T · · Systems · · ·



Benutzerhandbuch

Die in diesem Handbuch aufgeführten Produkte sind urheberrechtlich geschützt und stehen dem jeweiligen Rechtsinhaber zu.

rvsEVO
Version 4.0
Benutzerhandbuch

Pascalstraße 11
10587 Berlin

1 Einführung

Dieses Kapitel beinhaltet eine kurze Beschreibung der Produkte rvs® und rvsEVO, sowie eine Erklärung der typografischen Auszeichnungen, die in diesem Handbuch verwendet werden.

1.1 Kurze Beschreibung des Systems

Was ist rvs®

rvs® = Rechner-Verbund-System

Die Abkürzung rvs® steht für die Bezeichnung Rechner-Verbund-System. Das rvs® Rechner-Kommunikations-System ist ein etablierter Basisdienst für elektronischen Datenaustausch, EDI.

rvs® hat die Aufgabe, die Übertragung von elektronischen Daten zwischen heterogenen Computersystemen zu gewährleisten, die unterschiedliche Netzwerkprotokolle verwenden.

Um das zu erreichen, verwirklicht rvs® ein universelles Netzwerkmodell, das Sie für jeden Netzwerkknoten konfigurieren können.

rvs® sorgt für einen zuverlässigen und leistungsfähigen Transportdienst für standardisierte EDI-Nachrichten und für Dateien mit beliebigem Format und Inhalt. Sie können nur Dateien empfangen, die für rvs® vorgesehen sind. Das bedeutet, dass rvs® keinen unautorisierten Zugang zu anderen oder den eigenen Daten zulässt.

Das System wurde ursprünglich von der Volkswagen AG entwickelt und wird seit mehreren Jahren in der deutschen und europäischen Automobilindustrie genutzt, ist aber auch weltweit bei Banken, Versicherungen und in der Industrie im Einsatz.

rvs® arbeitet mit dem OFTP-Protokoll.

Was rvs® nicht ist

rvs® ist kein Onlinesystem. Es unterstützt weder den direkten terminalähnlichen Zugang zu anderen Rechnern, noch ermöglicht es eine Kommunikations-Pipe von Anwendung zu Anwendung auf Datensatz-Ebene. Sie können keine direkte Übertragung Ihrer Daten aus der Anwendung heraus ausführen. Sie können jedoch Sendeaufträge aus der Anwendung heraus an rvs® übergeben, die dann asynchron ausgeführt werden.

rvs® ist kein System, das Arbeitsaufgaben einplant.

rvs® interessiert sich nicht für den Inhalt der Dateien, die es überträgt. Es funktioniert als nachvollziehbares Transportmedium und führt keine Bedeutungsinterpretation der Daten durch, die es übermittelt.

rvs® ist kein EDI-Konverter. Jedoch sind zusätzliche Komponenten zur Konvertierung zwischen spezifischen Nachrichtenformaten (z.B. VDA, ODETTE, EDIFACT, XML), die rvs® als Transportdienst benutzen, bei der gedas deutschland GmbH verfügbar.

rvs® ist keine Software zur Netzwerksteuerung oder Überwachung.

Was ist rvsEVO

rvsEVO

rvsEVO ist eine Kommunikationssoftware mit grafischer Benutzeroberfläche, die wie rvs®, auf dem OFTP-Protokoll basiert. In der näheren Zukunft ist rvsEVO als Ablösung von rvs® geplant, unterstützt aber zurzeit noch nicht alle Funktionalitäten von rvs®.

rvsEVO ist in Java implementiert. Es existieren zwei Varianten von rvsEVO: rvsEVO Standard Edition und rvsEVO Tiny Edition. Den Unterschied entnehmen Sie bitte den Kapiteln 1. 2 und 1.3.

Zurzeit unterstützt rvsEVO das TCP/IP- und das TLS-Protokoll, es werden bald weitere Netzwerkprotokolle (ISDN, X.25 und XOT) folgen.

Nähere Informationen über unterstützte Plattformen entnehmen Sie bitte dem Dokument `$RVS_HOME\doku\liesmich.txt` (Release Notes).

Hinweis: Lesen Sie bitte das Kapitel 1.4 "Repräsentationsmittel" für eine Erklärung von `$RVS_HOME`.

1.2 rvsEVO Standard Edition

Folgende Funktionalitäten stehen bei rvsEVO Standard Edition 4.0 zur Verfügung:

Funktionalitäten

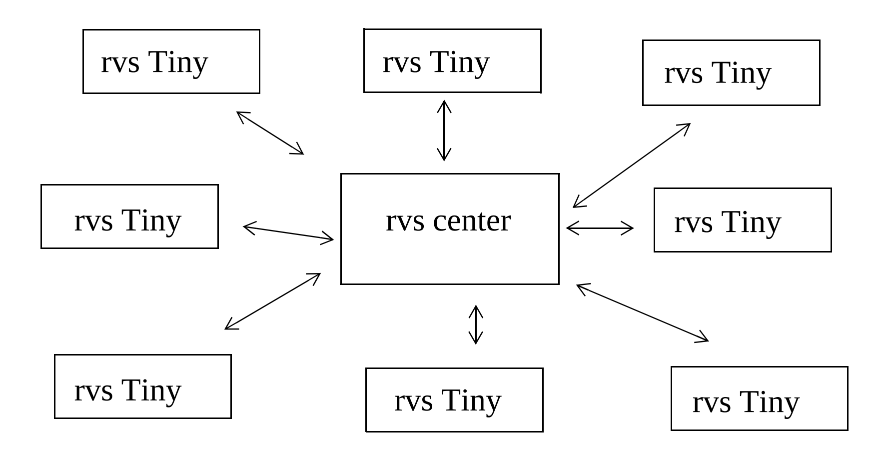
- grafische Benutzeroberfläche (GUI).
- Versand von Dateien an direkte Nachbar- oder an geroutete Stationen.
- Dateiempfang von direkten Nachbarstationen oder von gerouteten Stationen.
- Aktivierung der direkten Nachbarstationen, um abholbereite Dateien zu empfangen oder die Verbindung zu testen.
- Unterstützung von OFTP Version 1.3, 1.4 oder 2.0.
- Anzeige von Informationen über aktive Empfangs- und Sendeübertragungen, über beendete und über fehlerhafte Übertragungen.
- Verfolgung von Monitoraktivitäten und Unterstützung bei der Fehleranalyse mit Hilfe der rvsEVO-Logdateien.
- Löschen oder Freigeben von Empfangsbestätigungen (EERPs).
- Codeumwandlung (ASCII `->` EBCDIC und EBCDIC `->` ASCII) mit verschiedenen Umwandlungstabellen.
- Formatumwandlung während der Übertragung (unterstütze Formate: Fixed, Variabel, Text, Unstrukturiert).
- Jobstarts, um die gewünschten Aktionen beim Empfang oder Versand anzustoßen.
- Komprimierung und Verschlüsselung

– Backup und Recovery
– Zentrale Administration von anderen rvsEVO-Installationen.
– Unterstützung der Funktionen vom Zentralen Journal (Für mehr Informationen siehe Benutzerhandbuch Zentrales Journal)
– Unterstützung von SNMP-Monitoring (Für mehr Informationen siehe Benutzerhandbuch rvs®-SNMP-Agent).

rvsEVO kommuniziert mit der Applikation über eine Batch-Schnittstelle sowie über das Dateisystem. Die angebundene Applikation kann, sofern sie in der Lage ist, die erfolgreiche Weiterverarbeitung signalisieren und den erfolgreichen Versand signalisiert bekommen.

1.3 rvsEVO Tiny Edition

rvsEVO Tiny Edition ist eine schlanke Variante von rvsEVO, deren Hauptmerkmal ist, dass sie nur mit einer Nachbarstation (rvs® Zentrale) direkte Netzwerkverbindung hat. Dateien, die für andere Stationen gedacht sind, werden über die rvs® - Zentrale geroutet. Daher eignet sich diese Software besonders für eine Sterntopologie, in der eine Vielzahl von Filialen oder mobilen Mitarbeitern mit einem zentralen Server verbunden ist. Folgende Abbildung soll diesen Inhalt verdeutlichen:



Die Funktionalitäten von rvsEVO Tiny Edition werden durch den Lizenzschlüssel gesteuert.

Die folgenden weiteren Merkmale sind für rvsEVO Tiny Edition kennzeichnend:

– maximal zehn Partnerstationen
– 1 Nachbarstation (rvs® - Zentrale)

– maximal 2 Sessions
– nur eine Netzwerkkomponente (TCP/IP).

1.4 Repräsentationsmittel

Dieser Abschnitt enthält die Beschreibung, welche Ausprägungen und Auszeichnungen in diesem Handbuch verwendet werden und welche Bedeutung besonders gekennzeichnete Ausdrücke haben.

Typografische Auszeichnungen

- Handlungsanweisungen beginnen mit dem Punkt als Aufzählungszeichen.

– Sonstige Aufzählungen verwenden den Halbgeviertstrich.

Zeichenformate

`Courier`	Kommandos, Menübefehle, Dateinamen, Pfadnamen, Programme, Beispiele, Script-Dateien, Optionen, Qualifiers, Datensätze, Felder, Modi, Fensternamen, Dialogboxen und Status
FETT und **GROSSBUCHSTABIG**	Parameter, Umgebungsvariablen, Variablen
"Hochkommata"	Verweise auf andere Handbücher, Kapitel und Abschnitte, Literatur
Fett	wichtige Begriffe, Betriebssystemnamen, Eigennamen, Schaltflächen (Buttons), Funktionstasten.

Verzeichnisse

$RVS_HOME

Weil Benutzerverzeichnisse auf unterschiedlichen Plätzen bei den unterschiedlichen Betriebssystemen zu finden sind, benutzen wir in diesem Handbuch die Variable **$RVS_HOME**. Die Standardwerte sind:

– `C:\Programme\rvsEVO` für **Windows XP** und **Windows 2000**.

Ersetzen Sie diese Variable durch Ihren richtigen Pfad.

1.5 Zielgruppe

Dieses Handbuch ist sowohl für Benutzer gedacht, die mit rvsEVO routinemäßig arbeiten, als auch für Administratoren. Es gibt Ihnen einen Überblick über die Basisfunktionalität von rvsEVO.

Kenntnisse

Folgende Fähigkeiten sind erforderlich, um rvsEVO nutzen zu können:

– gute Kenntnisse über das benutzte Betriebssystem

– Kenntnisse über aktuell verwendete Kommunikationstechniken TCP/IP und TLS

Wir empfehlen dieses Handbuch zu lesen, bevor Sie anfangen mit rvsEVO zu arbeiten.

2 Installation

In diesem Kapitel werden die Installationsvoraussetzungen und der Installationsvorgang von rvsEVO beschrieben.

2.1 Systemvoraussetzungen

Für einen erfolgreichen Betrieb von rvsEVO benötigen Sie folgende Software:

Software

– Betriebssystem: Windows XP / Windows 2000, UNIX (AIX, SunOS, HP-UX, Linux) oder OpenVMS.
– Java Laufzeitumgebung (JRE) 1.4._XX oder Java Software Development Kit (JSDK) 1.4._XX.

  Falls auf ihrem System noch keine Java-Laufzeitumgebung installiert ist, dann holen Sie dies bitte vor der Installation von rvsEVO nach. Die Software ist frei verfügbar. Sie können diese über die Java Webseite *http://java.sun.com* herunterladen.

Am Anfang brauchen Sie mindestens 8 MB freien Platz auf Ihrer Festplatte. Abhängig von der Gebrauchsintensität, der Erhaltungsdauer für alte Einträge und der Zeitspanne zwischen Datenbankbereinigungen können die Speicherplatzanforderungen erheblich größer sein.

Zurzeit unterstützt rvsEVO das TCP/IP- und TLS-Protokoll. Weitere Netzwerke werden folgen (ISDN, X.25 und XOT).

In der Regel wird rvsEVO als CD ROM oder Datenband geliefert. Ihr System muss diese Datenträger lesen können. Wenden Sie sich bitte an Ihren Vertriebspartner (Telefon: +49 30 39971 537; Telefax: +49 30 39971 994; E-Mail: rvs-sales@gedas.de), wenn Sie andere Voraussetzungen haben.

2.2 Erhalten einer Lizenz

Zum Funktionieren benötigt rvsEVO einen Lizenzschlüssel.

rvs®-Kundendienst

Um einen Lizenzschlüssel zu erhalten, wenden Sie sich bitte an den rvs® - Kundendienst (Telefon: +49 30 39971 777; Telefax: +49 30 39971 994; E-Mail: rvs-service@gedas.de).

Folgende Schritte sind notwendig, um einen Lizenzschlüssel zu erwerben:

– In dem Eingabeaufforderungsfenster (`Ausführen -> cmd`) den Befehl `hostname` eingeben.
– Die Ausgabe dieses Befehls (Ihr Rechnername z.B. `BWcd00034`) ist dem rvs®-Kundendienst mitzuteilen. Beachten Sie dabei unbedingt die Groß-/Kleinschreibung, da andernfalls kein gültiger Lizenzschlüssel für Ihren Rechner generiert werden kann.

– Ihre Lizenzschlüsseldatei erhalten Sie per E-Mail.

Die Lizenzschlüsseldatei muss im Ordner `$RVS_HOME\conf\` unter dem Dateinamen `license.properties` gespeichert werden.

Die Komponenten für die Zentrale Administration (RCI und LCI, siehe Kapitel 8) werden extra lizenziert.

Hinweis: Lesen Sie bitte das Kapitel 1.4 "Repräsentationsmittel" für eine Erklärung von `$RVS_HOME`.

2.3 Neuinstallation von rvsEVO

Installationsschritte

In diesem Kapitel wird die Installation von rvsEVO beschrieben. Bevor Sie mit der Installation beginnen, vergewissern Sie sich, dass Ihre Systemumgebung alle Voraussetzungen für eine erfolgreiche Installation erfüllt (Siehe Kapitel 2.1 "Systemvoraussetzungen").

Unix-Systeme

Zuerst wird die Installation auf Windows-Systemen beschrieben. Anschließend wird kurz auf die Installation auf UNIX-Systemen eingegangen, da die Installationsschritte auf beiden Betriebssystemen gleich verlaufen.

OpenVMS

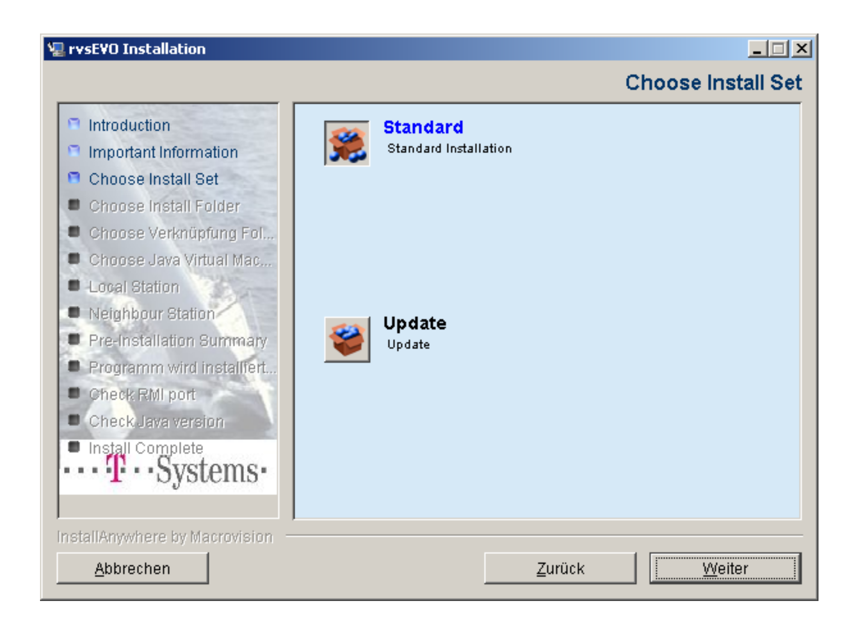
Die Installation auf OpenVMS-System ist in einem separaten Dokument beschrieben, das auf Anfrage bei Ihrem Vertriebspartner (Telefon: +49 30 39971 537; Telefax: +49 30 39971 994; E-Mail: rvs-sales@gedas.de) oder dem rvs® Kundendienst (Telefon: +49 30 39971 777; Telefax: +49 30 39971 994; E-Mail: rvs-service@gedas.de) zu erhalten ist.

Installation auf Windows-Systemen

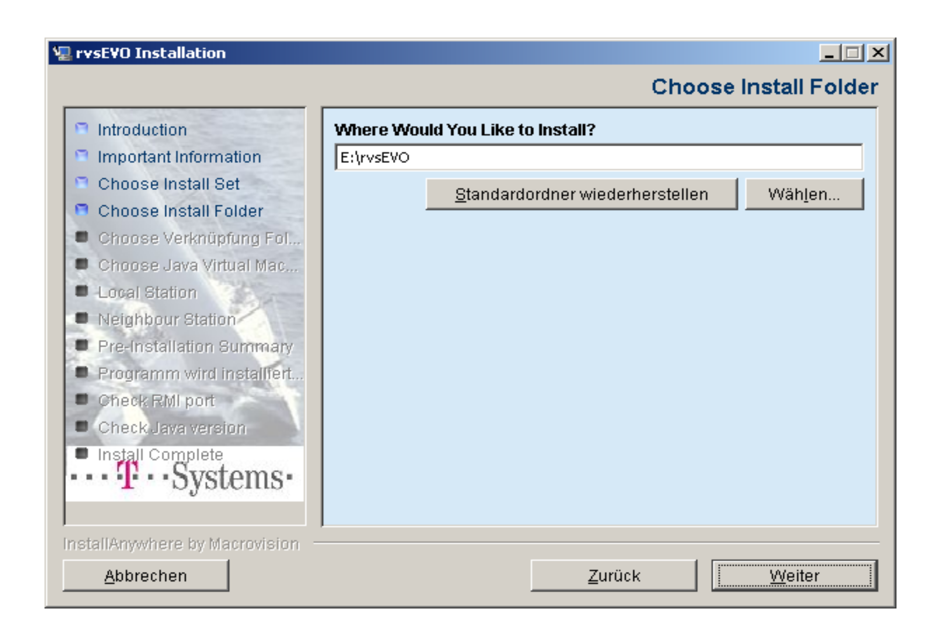
- Starten Sie Ihr Windows-System und melden Sie sich als Windows-Benutzer mit Administrator-Rechten an.
- Starten Sie die Installationssoftware `rvsEVO_X.X_setup.exe` (wobei `X.X` der Versionsnummer von rvsEVO entspricht) per Doppelklick oder über den Windows-Befehl: `Start -> Ausführen`.
- Im ersten Dialog können Sie die Sprachversion der Installation auswählen (Deutsch, Englisch, …). Mit **<OK>** kommen Sie zum nächsten Schritt der Installationsroutine.



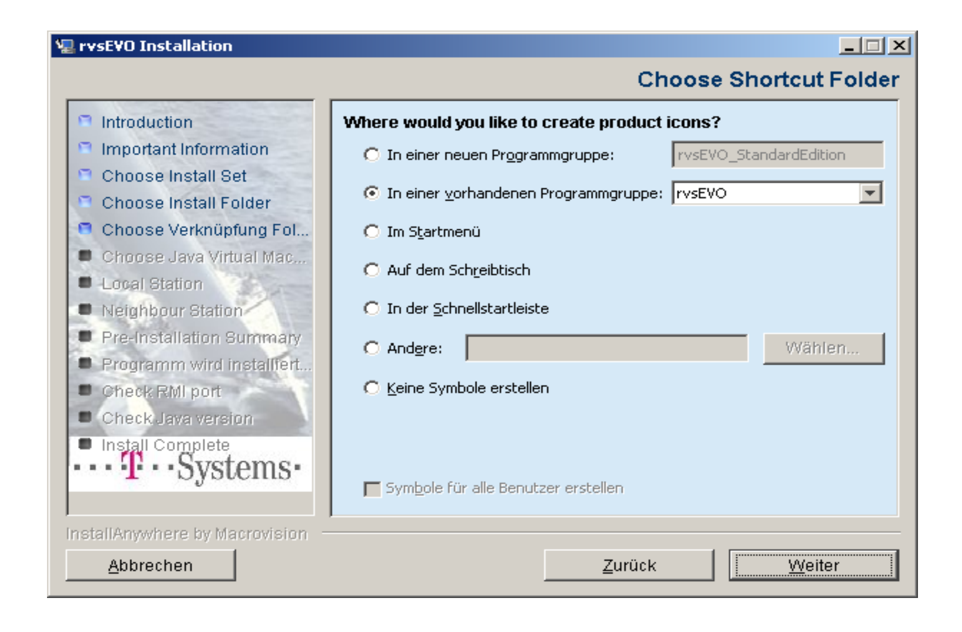
- Im nächsten Dialog können Sie zwischen einer „Standard“-Installation und einem Update wählen.



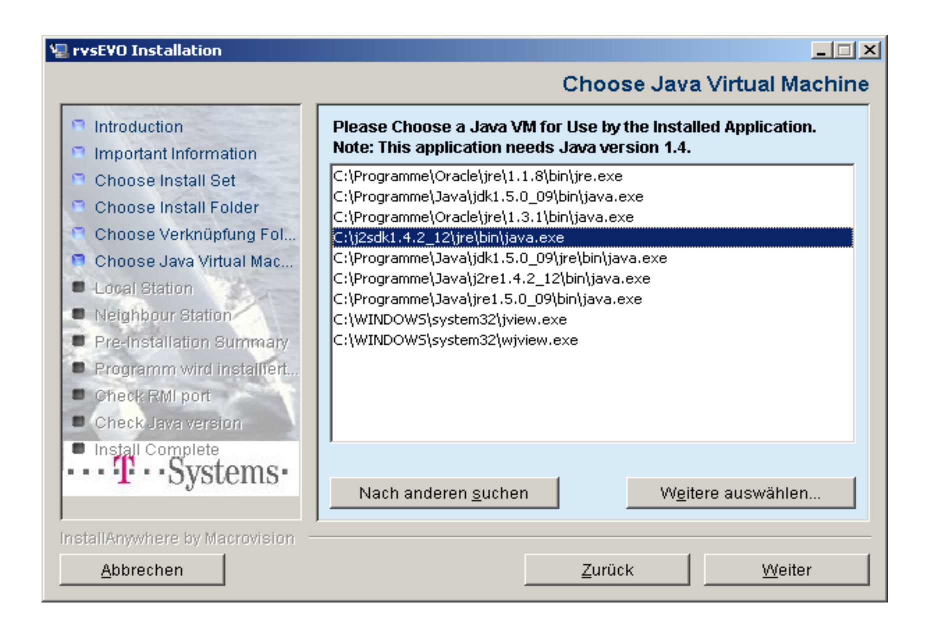
- Es folgt ein Dialog, in dem Sie das Verzeichnis auswählen können, in welches rvsEVO installiert werden soll.



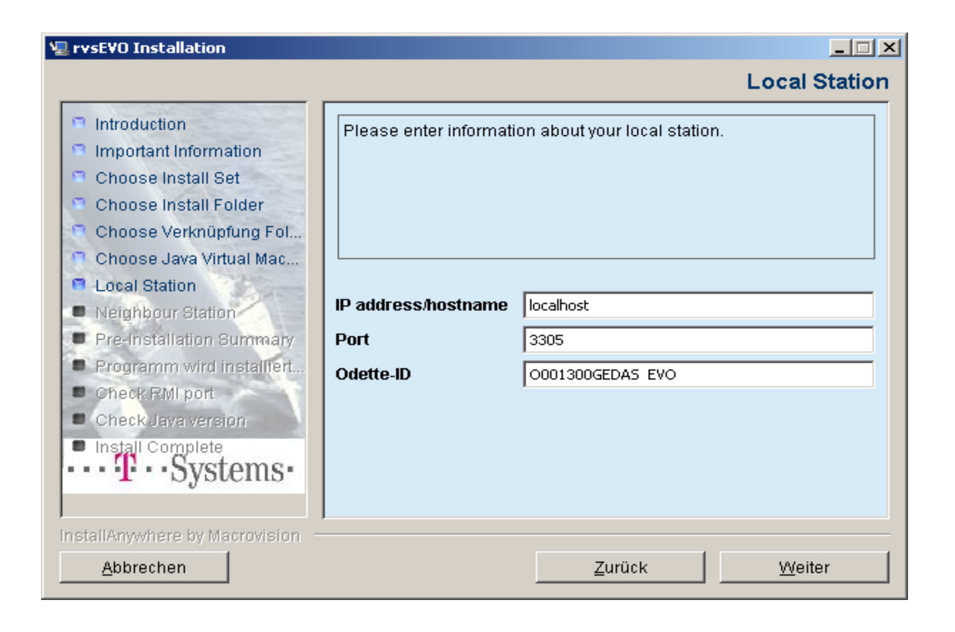
- Im dritten Dialog legen Sie fest, in welcher Programmgruppe die Symbole für rvsEVO erstellt werden sollen.



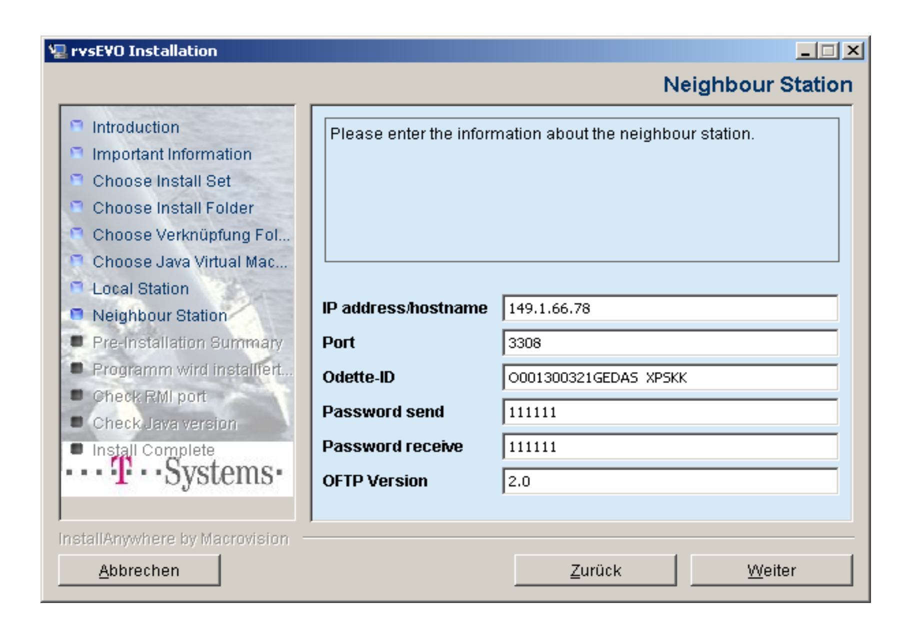
- Im nächsten Dialog wählen Sie aus, welche Java-Laufzeitumgebung für den Betrieb von rvsEVO eingesetzt werden soll. Hierbei sucht das Installationsprogramm nach installierten Komponenten und schlägt Ihnen die gefundenen Versionen in einem Auswahldialog vor. rvsEVO ist für einen Betrieb mit der Java-Version 1.4 freigegeben.



- Die folgenden Dialoge behandeln die Stationseinstellungen. Zunächst wird den Eintrag für die lokale Station konfiguriert. Die drei Parameter im Dialog sind obligatorisch und müssen eingegeben werden. Lesen Sie bitte das Kapitel 3.2 "Anpassen der Stationskonfiguration" für ausführliche Informationen über die Stationsparameter.



- Danach folgt der Dialog für die Eintragungen einer direkten Nachbarstation. Auch hier sind alle Parameter obligatorisch und müssen eingegeben werden, um reibungsloses Arbeiten von rvsEVO zu gewährleisten. Lesen Sie bitte das Kapitel 3.2 "Anpassen der Stationskonfiguration" für ausführliche Informationen über die Stationsparameter.



- Im nächsten Dialog gibt es noch eine kurze Zusammfassung der von Ihnen festgelegten Parameter (Installationsordner, Verknüpfungsordner). Gleichzeitig erhalten Sie Informationen über den erforderlichen und tatsächlich vorhandenen Speicherplatz. Durch Betätigen der Schaltfläche **Installieren** startet die Installation und kopiert die Installationsdateien in Ihre Verzeichnisse.
- Die letzte Bildschirmanzeige informiert Sie über die erfolgreiche Installation von rvsEVO.

Unix-Systeme

Installation auf Unix-Systemen

Wie schon am Anfang dieses Kapitels erwähnt, ist die Installation auf UNIX-Systemen analog zur Installation auf Windows-Systemen durchzuführen. Die Installationsdatei heißt `rvsEVO_X.X.X_setup.bin` und kann als Fensterinstallation unter dem X-Server oder im Konsolenmodus mit der Option `-console` gestartet werden.

Hinweis: Wenn Sie die Installation im Konsolenmodus durchführen, müssen Sie darauf achten, dass Sie die Installationsdatei als Shell-Skript aufrufen.

Beispiel (Aufruf):

```
sh ./rvsEVO_300_00_SE_setup.bin -console
```

Die Installationsabfragen in beiden Modi sind gleich (Siehe Abschnitt **Installation auf Windows-Systemen**).

Für bestimmte UNIX-Plattformen gibt es kleine Abweichungen, die in den Release Notes für die jeweilige Version (Dokument `$RVS_HOME\doku\liesmich.txt`) beschrieben sind.

2.4 Wie starte ich rvsEVO?

Windows: rvsEVO starten Sie mit Hilfe der rvsEVO-Programmgruppe: `Start -> Alle Programme -> rvsEVO -> rvs GUI.`

Unix: Starten Sie das Shell-Skript `$RVS_HOME/bin/startGUI.sh`

Mit diesem Menüpünkt (Programm) wird zuerst die Oberfläche gestartet, die anschließend den rvsEVO-Server startet.

Erfolgreicher Start:

Start

Ein erfolgreicher Start sieht dann folgendermaßen aus:



Zum Starten der Benutzeroberfläche dient das Programm `startGUI`, wohingegen zum Starten des rvsEVO-Servers das Programm `rvsservice` oder `startServer` auf Windows-Systemen und `startServer` auf UNIX-Systemen verwendet wird. Diese Programme befinden sich im Verzeichnis `$RVS_HOME/bin/` auf Windows-Systemen in Form von Batch-Dateien und auf UNIX-Systemen in Form von Shell-Skripten.

Hinweis für Windows:

Auf Windows-Systemen wird rvsEVO standardmäßig als Windows-Dienst gestartet. Dies ist in der Datei

$RVS_HOME/config/rvsConfig.xml mit dem Parameter RvsStartScript konfigurierbar (Siehe auch Kapitel 3.1).

Alternativ können Sie rvsEVO als Konsolenanwendung starten, indem Sie als RvsStartScript das Programm $RVS_HOME/bin/startServer.bat konfigurieren.

Unix: Auf Unix-Systemen wird standardmäßig als RvsStartScript das Shell-Skript $RVS_HOME/bin/startServer.sh gewählt.

2.5 Wie stoppe ich rvsEVO?

Windows: rvsEVO stoppen Sie mit Hilfe der rvsEVO-Programmgruppe: Start -> Alle Programme -> rvsEVO -> Stop Server.

Alternativ können Sie zum Stoppen von rvsEVO auf Windows-Systemen das Programm $RVS_HOME/bin/stopServer.bat aufrufen.

Unix: Zum Stoppen von rvsEVO wird das Shell-Skript $RVS_HOME/bin/stopServer.sh verwendet.

2.6 rvsEVO als Windows-Dienst

Windows-Dienst

rvsEVO wird standardmäßig als Windows-Dienst installiert (Siehe Kapitel 2.4).

Hinweis: Mit dem Begriff Dienst ist ein Programm gemeint, welches vom Betriebssystem aus gestartet werden kann und im Hintergrund arbeitet.

Die Installation von rvsEVO als Windows-Dienst ermöglicht Ihnen das Batch-Programm $RVS_HOME\bin\rvsservice.bat.

Syntax:

```
rvsservice <parameters> [options]
```

Parameter

Mögliche Parameter:

-c	startet rvsEVO in der Eingabeaufforderung
-i	installiert rvsEVO als Windows-Dienst
-r	deinstalliert rvsEVO als Windows-Dienst
-s	startet den Windows-Dienst rvsEVO
-h	Hilfe

Beispiel:

```
C:\Programme\rvsEVO\bin>rvsservice -i
---------------------------
-- RUSTINY SERVICE LAUNCHER --
---------------------------
rvs Server installed.
```

Mit dem Befehl `rvsservice -c` starten Sie den rvsEVO-Server in der Kommanozeile (Eingabeaufforderung).

Mit dem Befehl `rvsservice -i` installieren Sie rvsEVO als Windows-Dienst.

Startart

Nach der Installation von rvsEVO als Dienst können Sie rvsEVO in der Liste der Windows - Dienste finden (`Start -> Systemsteuerung -> Verwaltung -> Dienste`). Wenn Sie möchten, dass rvsEVO als Dienst bei jedem Systemstart automatisch gestartet wird, können Sie die Startart auf `Automatisch` setzen, indem Sie auf die Schaltfläche **Startart...**. klicken und die Startart `Automatisch` wählen.

Beispiel:



2.7 Update-Installation von rvsEVO

Update-Installation

Der Ablauf einer Update-Installation von rvsEVO ist fast identisch mit dem Ablauf einer normalen Installation (siehe Kapitel 2.3 "Neuinstallation von rvsEVO").

Lesen Sie bitte das Kapitel 8, um zu erfahren, wie Sie mit Hilfe der Zentralen Administration andere rvsEVO-Installationen updaten können.

3 Konfiguration

Dieses Kapitel beschreibt, wie Sie rvsEVO mittels der grafischen Benutzeroberfläche (GUI) oder mittels der XML-Konfigurationsdateien für Ihre Zwecke anpassen können.

Hinweis: Haben Sie alle Parameter während der Installationsvorganges richtig angegeben, brauchen sie keine Änderungen mehr an den Konfigurationsdateien vorzunehmen, um ein lauffähiges System zu erhalten. Nach jeder Änderung einer Konfigurationsdatei, ist ein neuer Start von rvsEVO notwendig.

3.1 Anpassen der globalen Parameter

Dateien

Es existieren zwei Konfigurationsdateien, die für die globalen Einstellungen von rvsEVO von besonderer Bedeutung sind:

– `$RVS_HOME/conf/rvsConfig.xml`

– `$RVS_HOME/conf/rvs.properties`.

Hinweis: In der Datei `rvsConfig.xml` befinden sich die rvsEVO-Parameter, die auch über die GUI konfigurierbar sind (`Admin -> Parameter`).

3.1.1 rvs.properties

Diese Datei ist von der Art her eine Java-Properties-Datei, die Einträge der Form:

`<name>=<value>`

enthält.

Diese Datei verweist auf 3 wichtige Orte (beinhaltet drei wichtigen Parameter):

`RootDir`

– das Installationsverzeichnis von rvsEVO (Parameter: `RootDir`)

`ConfigFile`

– die globale Konfigurationsdatei von rvsEVO (Parameter: `ConfigFile`)

`LogConfigFile`

– die globale Konfigurationsdatei für das System-Logging (Parameter: `LogConfigFile`)

Im Folgenden sehen Sie eine Beispieldatei (mit # beginnende Kommentarzeilen werden ignoriert):

```
RootDir=C:/Programme/rvsEVO
# configuration xml file
ConfigFile=C:/Programme/rvsEVO/conf/rvsConfig.xml
# log config file
LogConfigFile=C:/Programme/rvsEVO/conf/rvsLogger.xml
```

3.1.2 Anpassen der rvsEVO-Parameter

Parameter

Die rvsEVO-Parameter sind entweder:

– mittels der grafischen Benutzeroberfläche (`Admin -> Parameter`)



– oder mittels der XML-Konfigurationsdatei
`$RVS_HOME/conf/rvsConfig.xml`

konfigurierbar.

Es gibt die vier folgenden Parametergruppen:

– Environment: Dies sind allgemeine Parameter, die sich auf die rvsEVO-Umgebung beziehen.

– Notification: Diese Parametergruppe bezieht sich auf die Funktionalität rvs®-SNMP-Agent.

– Observer: Dies ist eine neue Funktionalität zum automatischen Durchsuchen von Verzeichnissen nach Sendeaufträgen. Sie entspricht dem Programm rvsjs in rvs® portable.

– Resource Check: Fehlerbehandlung in Bezug auf Ressourcen-Probleme.

Diese Parametergruppen sind entweder als Untermenüpunkte des Menüpunkts Parameter im Administration-Baum sichtbar oder als separate XML-Blocks in der Datei `rvsConfig.xml` zu finden.

rvsEVO-Umgebung (Environment)

In der folgenden Tabelle sind rvsEVO-Parameter, die sich auf die rvsEVO-Umgebung beziehen, aufgeführt.

PARAMETER	BESCHREIBUNG
BackupOnStartup	Mit diesem Parameter kann konfiguriert werden, dass bei jedem Start von rvsEVO eine automatische Sicherung durchgeführt wird. Mögliche Werte: Y (Yes); N (No): Standardwert ist Y. Wenn **BackupOnStartup** auf Y gesetzt ist, entsteht bei jedem Start von rvsEVO im Verzeichnis archive eine jar-Datei. In dieser jar-Datei werden folgende rvsEVO-Verzeichnisse gesichert: conf, jobs und system. Der Name der Sicherungsdatei besteht aus dem aktuellen Datum und Uhrzeit und der laufenden dreistelligen Numerierung. Beispiel: 070104141055000.jar ist die Sicherung, die am 04.01.07 um 14:10:55 Uhr entstanden ist.
Browser	Name des Systembrowsers, der aufgerufen wird, wenn HTML-Dateien vorliegen z.B. explorer
Cleanupdays	Angabe in Tagen für Archivierung von abgeschlossenen Jobs. Alle Jobs, die älter sind als die Zeitangabe in diesem Parameter, werden mit Hilfe des Programms archiveJobs (Siehe Kapitel 4.10) in die Datei RevisionLog.xml gespeichert (Siehe auch Parameter PersistenceArchive). Standard: 7 (Tage) **Hinweis**: Verwenden sie entweder Cleanupdays oder Cleanuptime.
Cleanuptime	Angabe in Stunden:Minuten: Sekunden für die Archivierung von abgeschlossenen Jobs (Format HH:MM:SS). Alle Jobs, die älter sind als die Zeitangabe in diesem Parameter, werden mit Hilfe des Programms archiveJobs (Siehe Kapitel 4.10) in die Datei RevisionLog.xml gespeichert (Siehe auch Parameter PersistenceArchive). **Hinweis**: Verwenden sie entweder Cleanupdays oder Cleanuptime.
Cleanupinterval	Zeitintervall in Minuten zwischen der Ausführung zweier Archivierungen. Standard: 1440 **Hinweis**: Dieser Parameter ist nicht vom Parameter Cleanuptime oder dem Parameter Cleanupdays abhängig.

PARAMETER	BESCHREIBUNG
CentralJournalInstance	Dieser Parameter bezieht sich auf die Funktionalität „Zentrales Journal“, welche in einem getrennten Handbuch beschrieben ist. **CentralJournalInstance** bedeutet die rvs®-Zielstation, an die die Journal-Dateien versendet werden. Diese Station muss in der rvsEVO-Stationsliste existieren. Lesen Sie bitte im Benutzerhandbuch „Zentrales Journal“ mehr über diese Funktion.
EngdatConfigFile	Konfigurationsdatei für das Modul Engdat (siehe Kapitel 7 für mehr Informationen)
JournalFilenamePrefix	Das Präfix des Journaldateinamens: Standard `TINY`. Lesen Sie bitte im Benutzerhandbuch „Zentrales Journal“ mehr über diese Funktion.
SendJournalInterval	Zeitintervall in Sekunden zwischen dem Versenden zweier Journaldateien an die rvs®-Zielstation (definiert durch den Parameter `CentralJournalInstance`). Wenn hier kein Wert oder 0 angegeben wird, wird keine Journaldatei versandt. Lesen Sie bitte im Benutzerhandbuch „Zentrales Journal“ mehr über diese Funktion.
DB	Verzeichnis für Jobverwaltung mit den Unterverzeichnissen `ENDED`, `FAILED`, `RCV` und `SND`. In den Unterverzeichnisse `RCV` und `SND` werden die temporären, nicht vollständig abgearbeiteten Jobs gespeichert. Ins Verzeichnis `FAILED` werden die misslungenen und ins Verzeichnis `ENDED` erfolgreich abgeschlossenen Jobs abgelegt. Diese Verzeichnisse sind auch in der rvsEVO-GUI sichtbar (`Transfer -> Transmissions`).
TEMP	Verzeichnis für temporäre Nutzung.
INBOX	Verzeichnis, in dem empfangene Dateien abgelegt werden.
OUTBOX	temporäres Verzeichnis für Versanddateien.
ARCDIR	Archiv-Verzeichnis, in dem z.B. die Datei für das Revisionslog abgelegt wird.
LOGDIR	Verzeichnis für die beiden Log-Dateien: `monlog.log` und `rvsservice.log`.
FirstLanguage	Erste Sprache für die GUI, wirksam erst nach neuem Start von rvsEVO.
HelpFile	Pfad der Hilfe-Datei.
JobstartConfigFile	Konfigurationsdatei für Jobstart.

PARAMETER	BESCHREIBUNG
StationsConfigFile	Stationskonfigurationsdatei: beinhaltet die Konfigurationsparameter der lokalen Station, der Nachbarstation und gerouteten Stationen.
MonlogStylesheet	xslt-Datei, um das Aussehen der Datei `monlog.log` zu beinflußen.
HostAllowFile	Konfigurationdatei, die DNS-Namen oder IP-Adressen von Hosts enthält, von denen rvsEVO Kommandos an den rvsEVO Server geschickt werden dürfen.
HostDenyFile	Konfigurationdatei, die DNS-Namen oder IP-Adressen von Hosts enthält, von denen keine rvsEVO Kommandos an den rvsEVO-Server geschickt werden dürfen.
ConnSetupFailWaitTime	Zeit in Millisekunden, die rvsEVO nach dem Scheitern einer Verbindung wartet, bevor es versucht, die Verbindung erneut aufzubauen.
JobAfterSECreation	Skript, das gestartet werden soll, sobald der Sendeauftrag erzeugt wurde. Dieser Parameter ist optional und enthält den Namen eines Skriptes, das im Verzeichnis `$RVS_HOME/bin` existieren muss. Siehe auch Kapitel 4.5 "Versand einer Datei" (`CreateSendJob`). Als erster Parameter wird diesem Skript die ÜbertragungsID übergeben.
LooptestNeighbourSID	ID der Station, über die der Looptest (Senden einer Datei an die eigene lokale Station) ausgeführt wird. Die Datei soll an die eigene Station versendet werden und rvsEVO leitet diese Datei über eine Nachbarstation wieder an die eigene lokale.
MaxSessions	Maximale Anzahl gleichzeitig laufender Empfängerprozesse für TCP/IP-Kommunikation. Standard 20; Maximum ist begrenzt durch System-Ressourcen.
MaxMonLogCount	Anzahl der Log-Dateien `$RVS_HOME/log/monlog.log`, die generiert werden können. Die Datei mit der höchsten Zahl ist die zuletzt generierte.
MaxMonLogSize	maximale Dateigröße der Log-Datei `monlog.log` (in Bytes)
MaxRevisionLogCount	Anzahl der Revision-Dateien `RevisionLog.xml`, die generiert werden können. Siehe Kapitel 4.10, um zu erfahren wie man eine Revision-Datei erzeugt. Siehe auch Parameter `PersistenceArchive` in dieser Tabelle.
MaxRevisionLogSize	maximale Dateigröße in Bytes der Revision-Datei `RevisionLog.xml`

PARAMETER	BESCHREIBUNG
ManagementConfig-File	Name der Konfigurationsdatei für die Zentrale Administration (Siehe Kapitel über die Zentrale Administration in diesem Handbuch).
OFTPTimeout	Time Out in Millisekunden auf der ODETTE-Ebene; Default: 300 000, kein Maximum.
PersistenceArchive	Name der Datei mit den Übertragungsdaten (Statistikdaten); Standard: `$RVS_HOME/archive/RevisionLog.xml`
RedoLog	legt fest, ob eine Redolog-Datei geschrieben wird. Mögliche Werte: `N` (Standard) Redolog-Datei wird nicht geschrieben `Y` Redolog-Datei wird geschrieben. Mehr über dieses Thema können Sie im Kapitel 5 nachlesen.
RMIServiceHost	Hostname in der RMI-Registry ; Standard: localhost. RMI steht für Remote Method Invocation und ist ein Protokoll, welches für interne Prozesskommunikation in Java benutzt wird.
RMIServiceName	Name des rvsEVO-Dienstes in der RMI-Registry: Standard: rvsEVO
RMIServicePort	Port-Nummer für die Kommunikation in der RMI-Registry; Standard: 3755
Timestamp	legt fest, ob eine Datei bei Empfang mit einem Zeitstempel im Dateinamen abgelegt wird oder nicht. Mögliche Werte: `N` (Standard) Zeitstempel wird nur eingefügt, wenn der Dateiname schon existiert; `Y` Dateiname wird immer mit einem Zeitstempel als Übertragungs-ID versehen.
RvsStartScript	Aufruf des Skriptes, welches rvsEVO startet. Standard: `$RVS_HOME/bin/startServer.bat` oder als Windows-Dienst: `$RVS_HOME/bin/rvsservice.bat -s`.
SessionAliveTimeout	Untätigkeitsdauer in Millisekunden, während der eine Verbindung noch als aktiv betrachtet wird; Standard 600 000, kein Maximum.
TraceItem	Parameter zum Einschalten von Fehlerverfolgung (Traces). Folgende Werte sind möglich: `O` (für Odette-Ebene), `L` (für Leitungsebene) und `C` (für Controller). Die Trace-Ausgabe erfolgt in die Datei `$RVS_HOME/log/trace.log`.

PARAMETER	BESCHREIBUNG
TransmissionFail-WaitTime	Zeit in Millisekunden für einen erneuten Start der Übertragung nach einem Misserfolg.

Hinweis: Mehr Information über das Zentrale Journal und den rvs®-SNMP-Agenten erhalten Sie in den jeweiligen Benutzerhandbüchern.

Notification

Die Parametergruppe Notification bezieht sich auf die Funktionalität des rvs®-SNMP-Agenten. Mehr Informationen über den rvs®-SNMP-Agenten erhalten Sie im rvs®-SNMP-Agent-Benutzerhandbuch.

PARAMETER	BESCHREIBUNG
AgentHostName	Dieser Parameter definiert, ob der rvs®-SNMP-Agent eingeschaltet ist oder nicht. Standard: `Y` (Yes). Mögliche Werte: `Y` (Yes) oder `N` (No).
AgentPort	Mit diesem Parameter wird definiert in welchen Abständen (in Sekunden) eine Heartbeat-Meldung von rvsEVO an die UDP-Adresse (AgentHostname + AgentPort) des Agenten gesendet wird.
AgentHeartbeatInter-vall	Rechnername (oder IP-Adresse) des Agenten. Standard: `localhost`.
AgentLogLevel	Dieser Parameter definiert, ob LogMeldungen von rvsEVO an den Agenten gesendet werden oder nicht. Mögliche Werte: 0, 1. `0`: keine LogMeldungen werden gesendet. `1`: alle LogMeldungen werden gesendet
AgentActive	IP-Port des Agenten. Standard: `3744`.

Observer

Die Funktionalität Observer dient zum Erzeugen von Sendeaufträgen für Dateien, die in ein konfigurierbares Verzeichnis gestellt werden. Der Observer untersucht in regelmäßigen Abständen das zu beobachtende Verzeichnis nach Dateinamen, die einem bestimmten Muster entsprechen. Wenn solche Dateien gefunden werden, werden aus ihnen Sendeaufträge erzeugt. Die Sendeoptionen für Sendeaufträge können konfiguriet werden.

Der Observer ist auch in der Lage, statt der Erzegung eines Sendeeintrags ein Skript (Shell-Skript auf UNIX-Systemen oder Batch-Datei auf Windows-Systemen) zu starten.

Hinweis: Es ist auch möglich, mehrere Observer zu konfigurieren, um z.B. mehrere Verzeichnisse parallel zu beobachten.

In der folgenden Tabelle sind die Parameter, die sich auf die Funktionalität Observer beziehen, beschrieben. Diese Parameter finden Sie in der GUI (`Admin -> Parameter -> Observer`) oder in der Konfigurationsdatei `rvsConfig.xml`, Block `Observer`)

PARAMETER	BESCHREIBUNG
Dir	Das Verzeichnis, was beobachtet werden soll.
Status	Dieser Parameter definiert, ob die Funktionalität Observer aktiv (eingeschaltet) ist oder nicht. Mögliche Werte: `ENABLED` (eingeschaltet) und `DISABLED` (ausgeschaltet).
Mask	Die Suchmaske (regulärer Ausdruck), die angewendet werden soll z.B. `MKL*`. `MKL*` bedeutet, dass nach Dateien, die mit `MKL` beginnen, gesucht wird. Wenn eine solche Datei gefunden wird, wird für sie ein Sendeauftrag erzeugt. Für Versand werden die Parameter aus dem Feld **SendEntry** verwendet (siehe diese Tabelle).
Time	Zeitraum zwischen zwei Prüfläufen in Sekunden.
SendEntry	Angaben zum Sendeauftrag, welcher ausgeführt werden soll. Für einen Sendeauftrag sind folgende Parameter möglich: `SID`, `VDSN`, `OutputFormat`, `RecordLength`, `InputCode`, `OutputCode`, `CodeTable`, `Serial`, `Disposition`, `Label`, `Compression`, `Encryption`. Welche Werte diese Parameter annehmen können, lesen Sie bitte im Kapitel „Versand einer Datei“, da diese Parameter auch vom Programm `createSendJob` oder in der GUI verwendet werden.

Parameter für Speicherplatzüberwachung

Diese Parametergruppe bezieht sich auf Speicherplatzüberprüfung (Ressourcen) von rvsEVO-Verzeichnissen. Mit dieser Funktionalität wird gewährleistet, dass ein rvsEVO-Operator rechtzeitig auf Speicherplatzmangel reagieren kann. Es werden alle aktiven Verzeichnisse von rvsEVO überprüft (Verzeichnisse, die in der Konfigurationsdatei `rvsConfig.xml` in den folgenden Variablen angegeben sind: `<DB>`, `<TEMP>`, `<INPUT>`, `<OUTPUT>`, `<ARCDIR>` und `<LOGDIR>`).

Im Falle von Speicherplatzproblemen erscheint im Monitor-Log eine Meldung, die besagt, in welchen rvsEVO-Verzeichnissen nicht ausreichend Speicherplatz vorhanden ist. Gleichzeitig kann eine E-Mail an den zuständigen Administrator gesendet werden.

Es gibt drei Stufen des Speichermangels:

– erste Stufe: es wird eine Warnung ausgegeben
– zweite Stufe: es werden alle Empfänger abgeschaltet, sodass kein Dateiempfang mehr möglich ist.
– dritte (kritische) Stufe: es wird alles abgeschaltet und rvsEVO beendet.

Für alle drei Stufen wird eine Meldung in der Log-Datei ausgegeben und es besteht gleichzeitig die Möglichkeit eine E-Mail an den zuständigen Administrator zu senden.

Folgende Parameter sind möglich:

PARAMETER	BESCHREIBUNG
DiskSpace	Anzahl von Bytes, die in jedem zu überprüfenden rvsEVO-Verzeichnissen frei sein muss, bevor eine Warnmeldung in die Monitor-Log ausgegeben wird. Standard: 150 000 000.
RcvStopDiskSpace	Anzahl von Bytes, die die in jedem zu überprüfenden rvsEVO-Verzeichnis frei sein muss. Ein Unterschreiten bewirkt, dass Empfängerprozesse gestoppt werden, sodass kein Dateiempfang mehr möglich ist. Standard: 120 000 000.
CriticalDiskSpace	Die Anzahl von Bytes, die in jedem zu überprüfenden rvsEVO-Verzeichnis frei sein muss. Mit diesem Parameter soll die kritische Grenze für Speicherplatzmangel angegeben werden. Das Unterschreiten dieser Grenze bewirkt, dass alle rvsEVO-Prozesse gestoppt und rvsEVO beendet wird. Gleichzeitig haben Sie die Möglichkeit, ein Skript ausführen zu lassen (siehe Feld `System` in dieser Tabelle). Standard: 100 000 000.
SuspendTime	für zukünftige Versionen
Time	Zeit in Sekunden zwischen zwei Ressourcenprüfungen
Mail	E-Mail-Adresse des Administrators, der informiert wird, wenn eine Grenze (Stufe) des Speicherplatzmangels überschritten wird.
System	Skript, welches ausgeführt werden soll, wenn die kritische Grenze (Stufe) erreicht wurde (siehe Parameter **CriticalDiskSpace**).

XML-Datei rvsConfig.xml

In diesem Abschnitt geben wir ein paar Hinweise bezüglich des Aufbaus der XML-Konfigurationsdatei `rvsConfig.xml`.

Die Parameter in der Datei `rvsConfig.xml` müssen innerhalb des jeweiligen XML-Elements angegeben werden. Dabei ist die XML-Syntax zu beachten (Anfangs-Tag, Inhalt, Ende-Tag).

Beispiel:

```
### Auszug aus rvsConfig.xml ###################
...
<Environment>
<DB>c:\Programme\rvsEVO\jobs</DB>
<TEMP>c:\Programme\rvsEVO\files\temp</TEMP>
<INBOX>c:\Programme\rvsEVO\files\inbox</INBOX>
<OUTBOX>c:\Programme\rvsEVO\files\inbox</OUTBOX>
<ARCDIR>c:\Programme\rvsEVO\archive</ARCDIR>
<JobstartConfigFile>rvsJobstart.xml</JobstartConfigFile>
<StationsConfigFile>rvsStationlist.xml</StationsConfigFile>
<MonlogStylesheet>MonlogStylesheet.xslt</MonlogStylesheet>
<PersistenceArchive>RevisionLog.xml</PersistenceArchive>
<HostAllowFile>host.allow</HostAllowFile>
<HostDenyFile> host.deny</HostDenyFile>
<RMIServiceName>rvsEVO</RMIServiceName>
<RMIServiceHost>localhost</RMIServiceHost>
</Environment>
...
########## rvsConfig.xml ####################
```

Die Änderungen in der GUI werden auch automatisch in die Datei `rvsConfig.xml` übernommen. Andersherum muss nach jeder Änderung der Datei `rvsConfig.xml` rvsEVO gestoppt und wieder gestartet werden, um Änderungen wirksam zu machen (siehe Kapitel 4.1 und 4.2).

Hinweise:

Sie können die Pfade für die Elemente `DB`, `TEMP`, `INBOX`, `LOGDIR` und `ARCDIR` selber festlegen und verändern.

Beispiel:

```
<DB>D:\rvsEVO\jobs</DB>
<TEMP>C:\Programme\rvsEVO\temp</TEMP>
```

Die Namen der Dateien `rvsJobstart.xml` und `rvsStationlist.xml` sind frei wählbar; sie müssen nur gültige XML-Dateien sein und sich im `conf`-Verzeichnis befinden. Das Gleiche gilt auch für Werte von Elementen `HostAllowFile` und `HostDenyFile`.

Beispiel (rvsConfig.xml):

```
<StationsConfigFile>stationen.xml</StationsConfigFile>
```

Die Datei `stationen.xml` ist eine Stationsliste im XML-Format mit den notwendigen rvsEVO-Parametern (siehe Kapitel 3.2) und befindet sich im `conf` Verzeichnis von rvsEVO.

Die Eintragungen für `RMISeviceName` und `RMIServiceHost` dienen für die interne Kommunikation und sollen nicht verändert werden.

3.2 Anpassen der Stationskonfiguration

Stationen

Die Konfiguration der Stationen ist auf zwei Wegen möglich:

- mittels der grafischen Benutzeroberfläche (GUI) oder
- mittels der XML-Stationskonfigurationsdatei.

Sie benötigen einen Eintrag in der Stationskonfiguration für Ihre lokale Station und für die Nachbarknoten, mit denen Sie die Dateien austauschen. Die obligatorischen Parameter für die beiden Stationsarten (lokale und Nachbarknoten) wurden schon bei der Installation abgefragt und die Stationskonfigurationsdatei

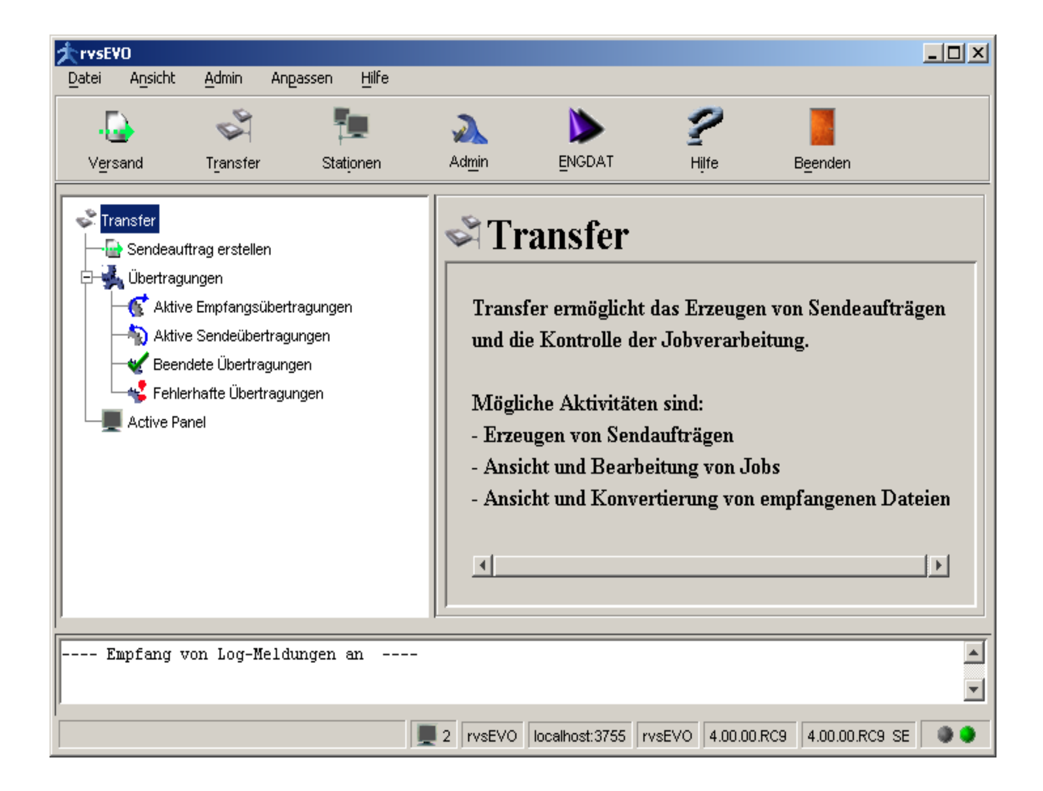
`$RVS_HOME/conf/rvsStationlist.xml`

wurde entsprechend angepasst. Diese Stationen sind dann auch nach dem Start der rvsEVO GUI sichtbar. Wie Sie rvsEVO starten können, lesen Sie bitte im Kapitel 4.1. Wenn Sie keine weiteren Stationen einrichten müssen, ist die Stationskonfiguration dadurch abgeschlossen und der Versand und Empfang von Dateien möglich.

3.2.1 Konfiguration von Stationen mittels GUI

Start

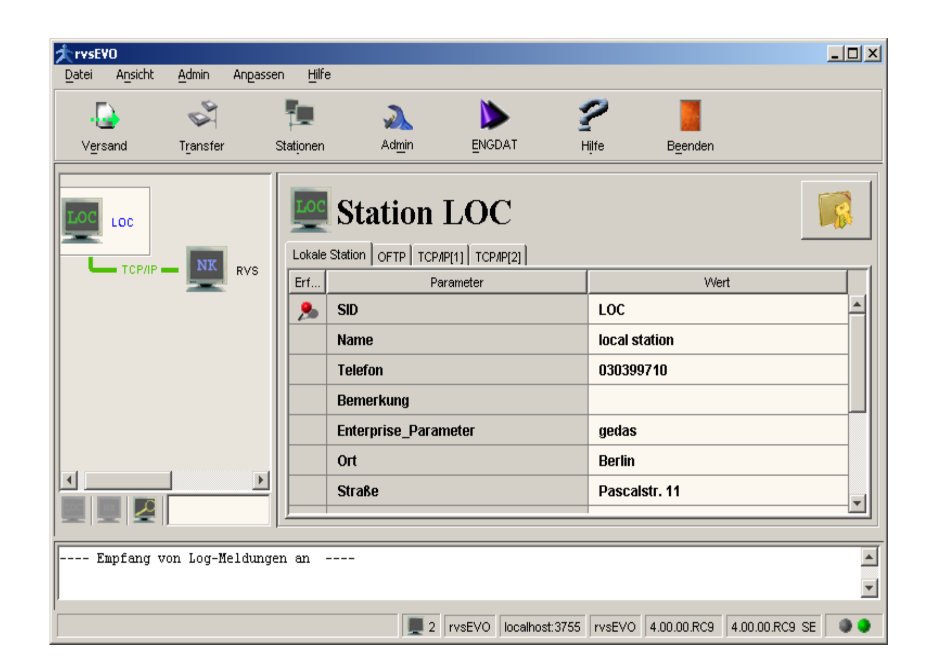
Nach dem Start von rvsEVO (Siehe Kapitel 2.4) erscheint das folgende Fenster:



Im oberen Bereich des Fensters sehen Sie eine Menüzeile (Datei, Ansicht, Admin, Anpassen, Hilfe) und darunter eine Funktionsleiste, in der Sie über die einzelnen Symbole die wichtigsten Funktionen (Versand, Transfer, Stationen, Admin, ENGDAT, Hilfe) aktivieren können.

Stationen

Um zu den Stationen zu gelangen, öffnen Sie das Fenster Stationen in der Funktionsleiste.



Im linken Bereich des Stationsfensters sehen Sie den Stationsbaum, im rechten Teil eine Stationsparametertabelle.

Der Stationsbaum stellt alle in der rvsEVO-Datenbank existierenden Stationen (Ihre lokale, die Nachbarknoten und die gerouteten Partnerstationen) dar. Per Mausklick können Sie eine der dargestellten Stationen markieren.

Registerkarten

In der Stationstabelle im rechten Fensterbereich werden Ihnen alle Parameter zur aktuell markierten Station angezeigt. Dabei können Sie mit Hilfe der verschiedenen Registerkarten unterschiedliche Parametergruppen konfigurieren.

Graue Felder deuten darauf hin, dass diese Parameter nicht editierbar sind.

Station löschen

Die Parameter, die für die Konfiguration einer Station obligatorisch sind, sind in der Spalte `Erfo...` (Erforderlich) mit dem Symbol gekennzeichnet.

Beispiel:

In der Registerkarte **OFTP** ist das `Odette Id`.

Lokale Station konfigurieren

lokale Station

Normalerweise werden die Parameter für das Einrichten einer lokalen Station schon bei der Installation festgelegt. In diesem Kapitel finden Sie eine ausführliche Erklärung aller für die lokalen Station möglichen Parameter.

Da die Konfiguration über die grafische Oberfläche oder mittels der XML-Konfigurationsdatei `rvsStations.xml` erfolgen kann, wird in der

Parametertabelle neben dem Parameternamen auch in spitzen Klammern der Name des XML-Elements angegeben, welches den Parameter repräsentiert.

Mögliche Registerkarten für die lokale rvsEVO-Station sind: Lokale Station, OFTP (Odette Parameter), TCP/IP, TLS.

– Lokale Station: der obligatorische Parameter in dieser Registekarte ist SID und Netzwerk. Der Parameter SID ist eine eindeutige PartnerID, die aus bis zu 16 Zeichen bestehen kann. Der Parameter Netzwerk wird automatisch angelegt, wenn Sie bei der Installation Ihre TCP/IP-Parameter eingeben. Die restlichen Parameter sind optional und dienen zur Pflege der Kontaktdaten.

– OFTP (ODETTE-Parameter): Für die lokale Station benötigen Sie nur ODETTE-ID einzutragen. Die ODETTE-ID ist eine weltweit eindeutige Identifikation aller Stationen, die das ODETTE-Dateiübertragungsprotokoll (OFTP) verwenden. Der Name hat 25 Zeichen, die folgende Verteilung aufweisen:

  – der Buchstabe O,

  – eine aus 18 Zeichen bestehende Organisationsidentifikation, die von der ODETTE Kodifikationsgruppe bereitgestellt wird, und

  – eine aus 6 Zeichen bestehende Unteradresse, die von jeder Organisation selbst vergeben wird.

  Hinweis: Wenn Sie nur innerhalb Ihres geschlossenen Netzwerks kommunizieren, können Sie die Länge der ODETTE-ID frei bestimmen, so dass sie in Ihrem Netzwerk eindeutig bleibt.

– TCP/IP-Parameter: In der nächsten Tabelle finden Sie die Erklärung der für das Netzwerk TCP/IP benötigten Parameter.

Parameter	Beschreibung
eingeschaltet **<enabled>**	Dieser Parameter entscheidet, ob der TCP/IP- Empfänger gestartet werden soll, oder nicht. Mögliche Werte: – Ja (eingeschaltet) – Nein (ausgeschaltet)
IP-Adresse **<IPAddress>**	IP-Adresse oder DNS-Name der eigenen Station. Wenn Sie bei der lokalen Station keinen Wert angeben, erlauben Sie die automatische Bestimmung der IP-Adresse. Wenn die eigene Station nur eine IP-Adresse besitzt, sollten Sie dieses Feld freilassen.
Port **<Port>**	Port auf dem ein TCP/IP-Listener lauschen soll; standardmäßig 3305

max. eingehende Verbindungen <Sessions>	Maximale Anzahl gleichzeitig laufender Empfangsprozesse über diesen Kanal. Maximum: 100
Empfänger-Nummer <ReceiverNumber>	**Nur bei der lokalen Station**: Nummer zur Unterscheidung der verschiedenen Empfangskanäle, über die die lokale Station erreichbar ist. Zu jeder Nummer gehört eine Registerkarte mit einem Satz von TCP/IP Parametern. rvsEVO vergibt und verwaltet diese Nummer automatisch. **Hinweis**: Dabei entspricht die Reihenfolge der Empfängernummern der Anordnung der Registerkarten in der grafischen Oberfläche. Die TCP/IP-Parameter in der ersten Registerkarte bekommen die Empfängernummer 1 (TCP/IP[1]); die TCP/IP-Parameter in der zweiten Registerkarte bekommen die Empfängernummer 2 (TCP/IP[2]) usw. Achten Sie beim Einrichten der Partnerstation darauf, über welche Empfängernummer die Kommunikation stattfinden soll. Siehe Parameter `Empfänger Nummer` (Partnerstation) in der Nachbarstation- Tabelle. Standard: **1**
Zeitüberschreitung <TimeOut>	Abbruchzeit (time out) in Sekunden, nach der das Kommunikationsprogramm die Verbindung abbricht, wenn die Partnerstation nicht antwortet.
Neustart Zeitintervall <RestartTimeout>	Zeitintervall in Sekunden, nachdem ein neuer TCP/IP-Listener starten soll.

Die TLS-Parameter werden benötigt, wenn die Kommunikation mit dem Partner verschlüsselt werden soll. TLS (Transport Layer Security) ist ein Verschlüsselungsprotokoll für Datenübertragungen im Internet. In der folgenden Tabelle sind viele Parameter gleich wie bei TCP/IP; es sind nur ein paar verschlüsselungsrelevante Parameter hinzugekommen.

Parameter	**Beschreibung**
eingeschaltet	Dieser Parameter entscheidet, ob der TLS- Empfänger gestartet werden soll, oder nicht. Mögliche Werte: – Ja (eingeschaltet) – Nein (ausgeschaltet)
IP-Adresse	IP-Adresse oder DNS-Name der eigenen Station für die TLS-Verbindung.
Port	Port auf dem ein TLS-Listener gestartet werden soll; standardmäßig 6619
max. eingehende Verbindungen	Maximale Anzahl gleichzeitig laufender Empfangsprozesse über diesen Kanal. Maximum: 100

Empfänger-Nummer	**Nur bei der lokalen Station**: Nummer zur Unterscheidung der verschiedenen Empfangskanäle, über die die lokale Station erreichbar ist. Zu jeder Nummer gehört eine Registerkarte mit einem Satz von TLS-Parametern. rvsEVO vergibt und verwaltet diese Nummer automatisch. **Hinweis**: Dabei entspricht die Reihenfolge der Empfängernummern der Anordnung der Registerkarten in der grafischen Oberfläche. Die TLS-Parameter in der ersten TLS-Registerkarte bekommen die Empfängernummer 1(TLS[1]); die TLS-Parameter in der zweiten TLS-Registerkarte bekommen die Empfängernummer 2 (TLS[2]) usw. Achten Sie beim Einrichten der Partnerstation darauf, über welche Empfängernummer die Kommunikation stattfinden soll. Siehe Parameter `Empfänger Nummer` (Partnerstation) in der Tabelle der Nachbarstation. Standard: **1**
Zeitüberschreitung	Abbruchzeit (time out) in Sekunden, nach der das Kommunikationsprogramm die Verbindung abbricht, wenn die Partnerstation nicht antwortet
Neustart Zeitintervall	Zeitintervall in Sekunden nachdem ein neuer TLS-Listener starten sollte
Schlüsseldateiname	Name der Schlüsselverwaltungsdatei für die TLS-Verbindung.
Schlüsseldateiname für vertrauenswürdige Zertifikate	Name der Schlüsselverwaltungsdatei für die vertrauenswürdigen Zertifikate (trust certificates).
client authentication	Bei einer TLS-Verbindung wird manchmal erwünscht, dass sich auch der Client (in diesem Fall, der Partner, von dem Sie die Daten empfangen), authentifiziert. Mit diesem Parameter können Sie festlegen, ob dies • nicht erforderlich • erwünscht oder • benötigt wird. **Hinweis**: Die Authentifizierung erfolgt mittels X.509-Zertifikaten.

Nachbarknoten konfigurieren

Nachbarknoten

Ein rechter Mausklick auf die lokale Station öffnet das Kontextmenü, in dem zwischen einer TCP/IP- und einer TLS-Verbindung gewählt werden kann. Eine TLS-Verbindung wird benötigt, wenn alle Daten, die über die Leitung gehen, verschlüsselt werden sollen.

Die möglichen Registerkarten für eine Nachbarstation sind: Nachbarstation, OFTP, Leitungstyp, TCP/IP und TLS.

– Nachbarstation: der obligatorische Parameter in dieser Registekarte ist SID. Der Parameter SID ist eine eindeutige PartnerID, die aus bis zu 16 Zeichen bestehen kann. Der Parameter Netzwerk wird automatisch angelegt, wenn Sie eine Nachbarstation anlegen, je nachdem für welche Verbindungsart Sie sich entscheiden (TCP/IP oder TLS). Die restlichen Parameter sind optional und dienen zur Pflege der Kontaktdaten.

– OFTP: Dieser Registerkarte beinhaltet Parameter, die sich auf das ODETTE-Protokoll beziehen. In der folgenden Tabelle werden Sie beschrieben.

Parameter	**Beschreibung**
Odette ID <OdetteID>	Im Gegensatz zur Stations-ID, die nur auf Ihrem Rechner eindeutig sein muss, ist die **ODETTE** ID eine weltweit eindeutige Identifikation der Partner- oder eigenen Station, wenn das ODETTE File Transfer Protokoll benutzt wird. Die **ODETTE** ID besteht aus 25 Zeichen, bestehend aus dem Buchstaben "O", der 18-stelligen Organisations-Identifikation und der 6-stelligen Computer-Adresse innerhalb der jeweiligen Organisation. Die Computer-Adresse ist von der Organisation frei wählbar und muss eindeutig sein.
Empfangspasswort <PasswordReceive>	Das Passwort, das rvsEVO von der Nachbarstation erwartet.
Sendepasswort <PasswordSend>	Das Passwort, das rvsEVO der Nachbarstation sendet. Die ODETTE-Passwörter werden immer zwischen den Nachbarstationen während des Aufbaus einer Session ausgetauscht und überprüft.
EERP ausgehend <EERPOut>	Handhabung für das Senden von Empfangsbestätigungen (EERPs). `NORMAL`: Generieren einer Empfangsbestätigung nach erfolgreichem Empfang einer Datei und unmittelbares, aktives Versenden. `HOLD`: Generieren einer Empfangsbestätigung nach erfolgreichem Empfang einer Datei. Empfangsbestätigung wird jedoch erst nach Freigabe mit dem Programm `handleEERP` versendet. Default: `NORMAL`
Austauschpuffergröße <BufferSize>	Maximale Größe des Übertragungspuffers (Exchange Buffer Size) in Bytes. Mögliche Werte: 0 - 99999.
Austauschpuffer-Credit <Credit>	Maximale Zahl der gesendeten Blöcke (Exchange Buffer) ohne Erwartung einer Quittung. Mögliche Werte: 0 - 999.

OFTP-Version <level>	Hier ist die anzuwendende ODETTE-Protokollversion festzulegen. Normalerweise muss ein OFTP-Produkt in der Lage sein, während einer Odette-Session die Odette-Protokollversion auszuhandeln. Wenn dies Ihr Partner nicht kann, sollen Sie die ODETTE-Protokollversion über diesen Parameter fest einstellen. Mögliche Werte : 1.2,1.3, 1.4, 2. Standard: **2**

– Leitungstyp: In der Registerkarte **Leitungstyp** befindet sich nur der Parameter `Aktiver Verbindungsaufbau`. Mit diesem Parameter können Sie festelegen, ob Ihre lokale Station zu der in diesem Parametersatz vereinbarten Station aktiv die Verbindung aufbauen soll oder nicht. `Ja` bedeutet, dass sobald die Sendeaufträge erzeugt werden, die Verbindung zur jeweiligen Station automatisch aufgebaut wird und Dateien versendet werden; bei `Nein` werden anstehende Sendeauträge nicht automatisch versendet, sondern es wird gewartet, bis die Gegenstelle die Verbindung aktiviert. Erst wenn die Verbindung steht, werden Dateien versendet.

– TCP/IP und TLS: Die TCP/IP und die TLS-Registerkarte haben die gleichen Parameter.

Parameter	Beschreibung
IP-Adresse	IP-Adresse oder DNS-Name der eigenen Station für die TCP/IP- oder TLS-Verbindung.
Port	Port auf dem ein Listener gestartet werden soll; standardmäßig 6619
Nummer des zugehörigen Empfängers	Nummer des Empfangskanals, wenn für die lokale Station mehrere Empfängers konfiguriert wurden (mehrere Registerkarten). Über diese Nummer ist es möglich festzulegen, über welche Empfänger-Nummer die Kommunikation für die betreffende Station stattfindet. Die TCP/IP- und TLS-Empfänger werden separat hochgezählt. Wenn für die lokale Station nur ein Empfänger definiert wurde, ist dies die `Empfänger Nummer 1` (Standard)

Ein rechter Mausklick auf einen Nachbarknoten (eine Nachbarstation) öffnet das Kontextmenü mit den Optionen: **geroutete Station hinzufügen, Eintrag entfernen** und **Verbindung aktivieren**. Mit der Option **Verbindung aktivieren** können Sie die Nachbarstation aktivieren, um die Dateien, die für Ihre Station bestimmt sind, abzuholen. Mit der Option **geroutete Station hinzufügen** können Sie eine geroutete Station anlegen, die über die direkte Nachbarstation erreichbar ist. Mit dem **Eintrag entfernen** können Sie eine schon angelegte geroutete Station löschen.

Eine geroutete Station anlegen

geroutete Station

Voraussetzung: Sie haben einen direkten Nachbarknoten, über den Sie diese geroutete Station erreichen können, schon eingerichtet.

Sie klicken mit der rechten Maustaste auf die Nachbarstation und wählen im Kontextmenü den Eintrag **geroutete Station hinzufügen**. Ein neuer Pfeil zeigt auf die neue Station. Für Sie ist es nicht von Bedeutung, mit welcher Verbindungsart diese Station erreicht wird (das regelt die Nachbarstation). Deswegen ist auf diesem Pfeil auch keine Verbindungsart zu sehen.

Die möglichen Registerkarten für eine geroutete Station sind: Geroutete Station und OFTP.

Die obligatorischen Parameter in der Registerkarte Geroutete Station sind SID und Nachbarstation.

– SID: StationID der gerouteten Station
– Nachbarstation: Eine geroutete Station kann nur über eine direkte Nachbarstation erreicht werden, da es zu einer gerouteten Station keine direkte Netzwerkverbindung gibt. Die Information über welche Nachbarstation diese geroutete Station zu erreichen ist, ist in diesem Parameter hinterlegt.

Dieser Parameter entspricht dem Parameter `Gateway` im Element `StationRouted` in der XML-Stationskonfigurationdatei.

In der Registerkarte OFTP sind die ODETTE-Parameter. Der obligatorische Parameter in dieser Registerkarte ist nur `Odette Id`. Lesen Sie bitte den Abschnitt **Nachbarknoten konfigurieren** in diesem Kapitel für die Erklärung der Parameter `Odette Id`, `EERP ausgehend` und `OFTP-Version`.

Eine Station löschen können Sie mit der Option **Station entfernen.** Zu dieser Option gelangen Sie, indem Sie mit einem Klick der rechten Maustaste das Kontextmenü einer markierten Station öffnen.

3.2.2 Konfiguration von Stationen mittels der XML-Datei

Alternativ zur grafischen Oberfläche kann die Stationskonfiguration auch in der XML-Stationskonfigurationsdatei `rvsStationlist.xml` durchgeführt werden. Das Element `StationLoc` aus der Stationskonfigurationsdatei ist äquivalent zu der Lokalen Station in der GUI, `StationNeighbour` ist die Nachbarstation und `StationRouted` die Geroutete Station.

Auszug aus der Datei `rvsStationlist.xml`:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<RVS_STATION_CONFIG>
<!-- Einstellungen für die Lokale Station -->
<StationLoc>
 . . .
</StationLoc>
<!--Einstellungen für die Nachbarstation -->
```

```
<StationNeighbour>
 . . .
<StationNeighbour>
<!----Einstellungen für geroutete Stationen  -->
<!--Für jede geroutete Station ein Eintrag -->
<StationRouted>
. . .
</StationRouted>
</RVS_STATION_CONFIG>
```

Kommentare

Bereiche, die mit dieser Zeichenfolge (<!--) beginnen und dieser Zeichenfolge (-->) enden, werden als Kommentar interpretiert.

Benutzen Sie zum Editieren der Datei ein Textbearbeitungsprogramm (z.B. `Edit`, `TextPad`). Achten Sie bitte bei der Bearbeitung der XML-Dateien darauf, dass Sie diese als gültiges XML-Dokument ablegen, sonst kann rvsEVO diese nicht lesen und eventuell nicht richtig starten.

Hinweis: Sie müssen dafür sorgen, dass die IP-Ports: RMI-Port (1099) und der Odette-Port (z.B. `3305`) frei sind, damit die TCP/IP-Kommunikation einwandfrei funktioniert.

Die Stationsparameter sind schon in den Kapitel 3.2.1 beschrieben. Der Name des XML-Elements, das den Parameter repräsentiert wurde immer in spitzen Klammern aufgeführt.

3.3 Konfiguration der Jobstarts

Jobstarts

Die Jobstarts beinhalten Regeln, die dazu führen, dass bei entsprechenden ein- bzw. ausgehenden Dateien spezielle Programme gestartet werden können. Sie entsprechen den Residenten Empfangseinträgen (Jobstarts nach Empfang) und den Jobstarts nach Sendeversuchen in rvs®.

Jobfilter

Falls mehr als ein Jobfilter auf den Sende- bzw. Empfangsjob zutrifft, so werden alle Programme der zutreffenden Jobfilter gestartet, wobei die Reihenfolge nicht festgelegt werden kann.

Die Jobstarts können über die rvsEVO-GUI oder in der XML-Jobstarts-Konfigurationsdatei `rvsJobstart.xml` konfiguriert werden.

Konfiguration der Jobstarts mittels GUI

GUI

Die Jobstarts werden im Administrationsfenster konfiguriert. Zu diesem Fenster gelangen Sie, indem Sie das Symbol Admin in der Funktionsleiste anklicken.

Wenn Sie sich im Admin-Fenster befinden, können Sie zwischen Jobstarts nach Empfang und Jobstarts nach Sendeversuch wählen, je nachdem, ob Sie eine bestimme Aktion gekoppelt an den Empfang oder an den Versand von Dateien auslösen möchten.

Einen neuen Jobstart können Sie anlegen, indem Sie im Admin-Baum `Jobstarts` markieren und mit der rechten Maustaste den Menüpunkt **Neuen Eintrag erstellen** wählen. Sie können dann im neu erstellten

Eintrag bei der Richtung festlegen, ob dies ein Jobstart nach Empfang oder ein Jobstart nach Sendeversuch werden soll. Die zweite Möglichkeit ist, sich sofort für die Richtung zu entscheiden, indem Sie Jobstart nach Empfang oder Jobstart nach Sendeversuch markieren und anschließend mit der rechten Maustaste den Menüpunkt **Einen neuen Eintrag erstellen** wählen.

Nachdem Sie einen neuen Jobstarteintrag erstellt haben, bieten sich in der Jobstartmaske folgende Filter an: Richtung (Direction), SID, VDSN, Prozess, Shell und Sendversuche (nur beim Jobstart nach Sendeversuch). Lesen Sie bitte detaillierter zu den Filtern in der Tabelle im nächsten Abschnitt.

Konfiguration der Jobstarts in der XML-Datei

Die XML-Konfigurationsdatei `rvsJobstart.xml` dient der Konfiguration der Jobstarts.

Diese Datei ist wie alle anderen rvsEVO - Konfigurationsdateien auch im XML-Format.

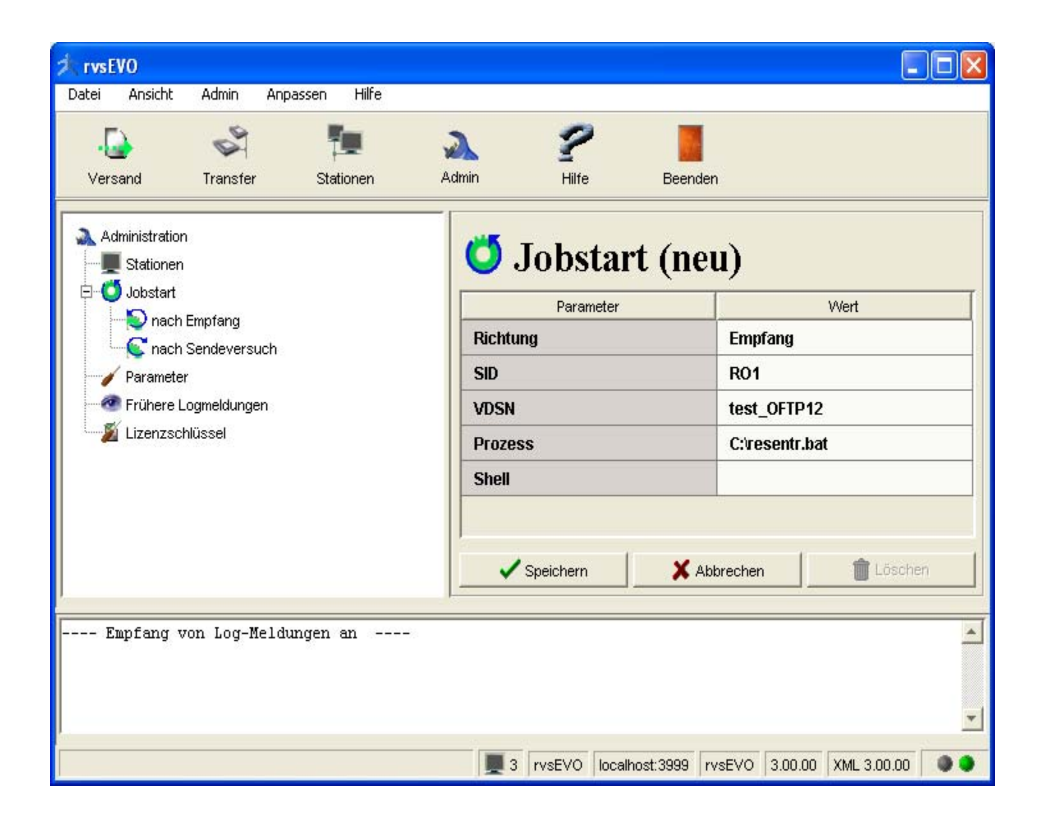
```
<jobstarterData>
       <jobfilters>
              <jobfilter>
                     <vdsn></vdsn>
                     <sid></sid>
                     <direction>SND</direction>
                     <sendAttempts>0</sendAttempts>
                     <process>C:\jobstart.bat</process>
              </jobfilter>
              <jobfilter>
               ...
              </jobfilter>
              ...
        </jobfilters>
</jobstarterData>
```

Diese Datei beinhaltet beliebig viele Jobfilter-Elemente. Die einzelnen Unterelemente von Jobfilter werden in der folgenden Tabelle genau beschrieben.

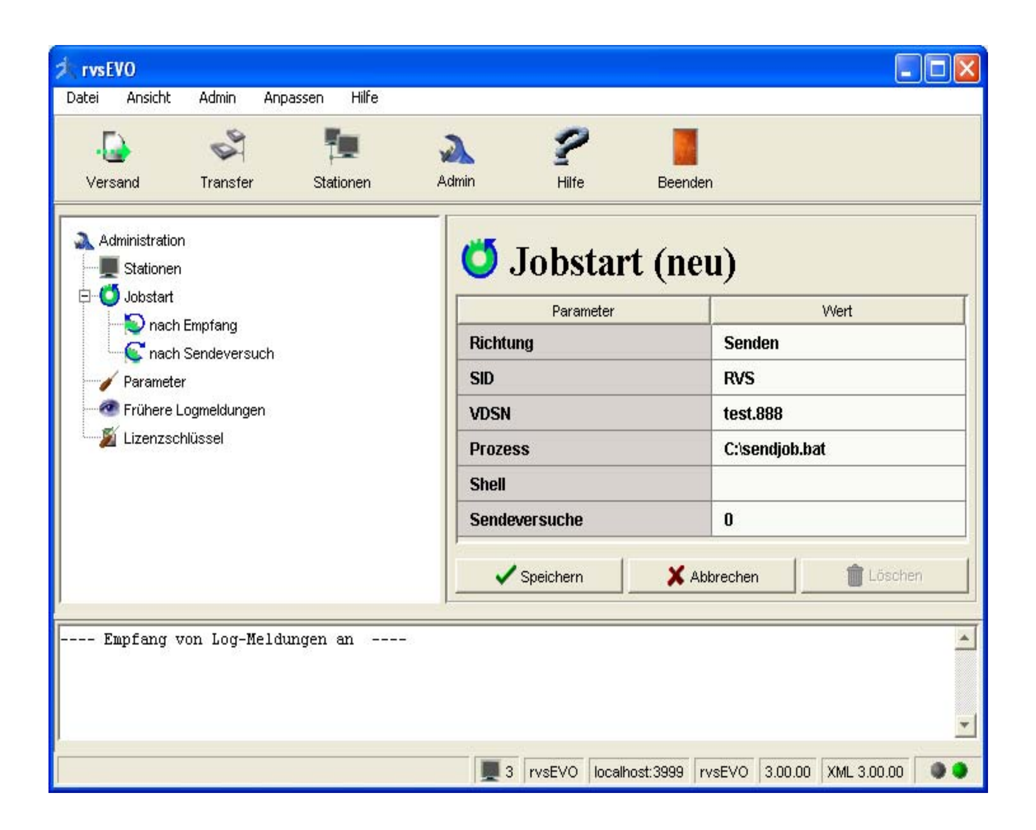
vdsn	Virtueller Dateiname (Regulärer Ausdruck als Filter), max.Länge 26 Zeichen.
sid	Stations-Id (Regulärer Ausdruck als Filter für die StationsId)
direction	Legt Filterbedingung für die Richtung der Kommunikation fest. Mögliche Werte: `SND` (Bei Dateiversand), `RCV` (Bei Dateiempfang).
sendAttempts	Anzahl der fehlgeschlagenen Sendeversuche. Wird hier "0" eingetragen, so bedeutet dies erfolgreicher Dateiversand.

process (mandatory)	Programm, das gestartet werden soll, wenn die Filterbedingungen alle zutreffen. Den Programmen wird ein fest definierter Satz von Parametern übergeben. Parameter: 1. jobId 2. Stations ID (des Senders bzw. des Empfängers) 3. Dateiname der gesendeten/bzw. empf. Datei 4. VDSN 5. Datum und Uhrzeit des Jobs 6. Anzahl der Sendeversuche.
Shell	Kommando-Shell, in der das Programm ausgeführt werden soll; z.B. `cmd` (Standard bei Windows-Systemen; `ksh`,`.csh`, ...).

Beispiel (GUI) Beim Empfang der Datei mit dem virtuellen Dateinamen `test_OFTP12` von der gerouteten Station `RO1`, startet rvsEVO das Programm `C:\resentr.bat`.



Beispiel (GUI) Wenn rvsEVO eine Datei mit dem Namen `test.888` an die Nachbarstation `RVS` erfolgreich (SendAttempts=0) versendet, startet das Programm `C:\sendjob`.



Jobstarteintrag löschen

Wenn Sie einen Jobstarteintrag löschen möchten, müssen Sie ihn zuerst mit Doppelklick auswählen. Im neuen Fenster haben Sie dann die Möglicheit, den ausgewählten Jobstart mit der Schaltfläche **Löschen** zu entfernen.

4 Arbeiten mit rvsEVO

Batch-Dateien

In diesem Kapitel werden alle Programme, die Ihnen für den täglichen Umgang mit rvsEVO zur Verfügung stehen, beschrieben. Diese Programme befinden sich als Batch-Dateien (Windows) oder Skripte (UNIX) im Verzeichnis `$RVS_HOME\bin`.

Hinweis: Sie müssen sich im Verzeichnis `$RVS_HOME\bin` befinden, wenn Sie ein rvsEVO Programm aufrufen.

Manche von den rvsEVO-Programmen (wie z.B. GUI, Start und Stopp von rvsEVO) stehen auch über die rvsEVO-Programmgruppe zur Verfügung (`Start -> Alle Programme -> rvsEVO`).

4.1 Starten des rvsEVO-Servers

`startServer`

Mit dem Programm `startServer` können Sie den rvsEVO-Server in der Eingabeaufforderung starten.

Beispiel:

```
startServer
```

Es können keine Parameter übergeben werden. Ein erfolgreicher Start sieht folgendermaßen aus:

```
*

* rvs Server has started.

*
```

Hinweis: Es ist auch möglich, den rvsEVO Server über die rvsEVO - Programmgruppe zu starten (`Start -> Programme -> rvsEVO -> rvsEVO GUI`). Dabei wird zuerst das Programm startGUI gestartet, welches seinerseits den rvsEVO-Server startet.

4.2 Stoppen des rvsEVO-Servers

`stopServer`

Stoppen Sie rvsEVO mit dem Programm `stopServer`.

Syntax:

```
stopServer -m <mode> [-verbose]
```

Alle Parameter sind optional:

-m <mode>	Zeit für Beendigung der Jobs, bevor rvsEVO gestoppt wird. Mögliche Werte: `0` (default; 120 Sekunden), `1` (60 Sekunden), `2` (30 Sekunden), `3` (20 Sekunden), `4` (10 Sekunden).
-verbose	Ausgabe von ausführlichen Meldungen.
-help	Fordert Hilfe (Usage) an.
-?	Fordert Hilfe (Usage) an.

Beispiel:

```
stopServer
```

Ergebnis: Der Server wird in 120 Sekunden gestoppt.

Beispiel:

```
stopServer -m 3
```

Ergebnis: Der Server wird in 20 Sekunden gestoppt.

```
*
* rvs Server has stopped.
*
```

Hinweis: Es ist auch möglich, den rvsEVO Server über die rvsEVO - Programmgruppe zu stoppen (`Start -> Programme -> rvsEVO -> stop Server`).

4.3 Meldungen des Monitors anzeigen

rvsEVO bietet im unteren Teil der Benutzeroberfläche die Möglichkeit, aktuelle Log-Meldungen zu verfolgen.

Frühere Meldungen des Monitors können Sie über die GUI oder mit dem Programm `showMonitorLog` anzeigen lassen.

GUI

Logmeldungen

Die früheren Meldungen des Monitors können im Admin-Fenster im Baum Administration, Menüpunkt **Frühere Logmeldungen** angezeigt werden. Zum Admin-Fenster gelangen Sie, wenn Sie das Symbol Admin

in der Funktionsleiste anklicken. Wie Sie die rvsEVO-GUI starten können, lesen Sie bitte in Kapitel 2.4.



Eingabeaufforderung (Kommandozeile)

`show MonitorLog`

Benutzen Sie das Programm `showMonitorLog` um die Meldungen des Monitors zu verfolgen und eventuell Fehlermeldungen zu analysieren.

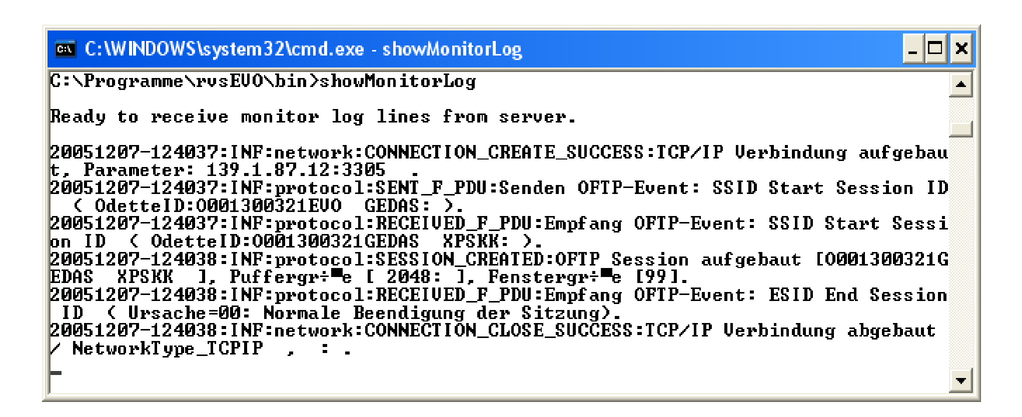
Syntax:

```
showMonitorLog
```

Optionale Parameter:

-verbose	Ausgabe von ausführlichen Meldungen.
-help	Gibt Beschreibung zum aktuellen Kommando aus
-?	Fordert Hilfe an.

Beispiel:



Die Meldungen des Monitors werden in die Datei `$RVS_HOME\rvsEVO\log\monlog.log` geschrieben.

Eine ähnliche Funktionalität wie das Programm `showMonitorLog` bietet auch das Programm `showMonitorLogFile`. Dabei kann eine Monitor-Log-Datei im Text- oder HTML-Modus betrachtet werden. Standardmäßig wird die Datei `monlog.log` genommen; wenn eine andere Mon-Log-Datei betrachtet werden sollte, muss dies mit der Option `-i` angegeben werden.

Syntax:

```
showMonitorLogFile [-TEXTMODE|-HTMLMODE] [-i]
```

Optionale Parameter:

-TEXTMODE	Ausgabe im Text-Format.
-HTMLMODE	Ausgabe im HTML-Format
-i	Monitor-Log-Dateiname; es muss ein existierende Dateiname sein.
-help	Gibt Beschreibung zum aktuellen Kommando aus
-?	Fordert Hilfe an.

4.4 Aktivieren einer Station

`activate Station`

Mit dem Programm `activateStation` können Sie eine direkte Nachbarstation aktivieren, um die Dateien, die für Ihre Station bestimmt sind, abzuholen (siehe auch Kapitel 3.2.1).

Syntax:

```
activateStation [-verbose]
```

Optionale Parameter:

-verbose	Ausgabe von ausführlichen Meldungen.
-help	Gibt Beschreibung zum aktuellen Kommando aus.
-?	Fordert Hilfe an.

Wenn die Aktivierung der Nachbarstation nicht erfolgreich sein sollte (wie z.B. wegen einer falschen IP-Adresse), müssen Sie auf die Nachrichten in dem Eingabeaufforderungsfenster, die mit Meldung oder Error beginnen, achten.

Im Falle einer falschen IP-Adresse, muss die Konfigurationsdatei mit der Stationsliste korrigiert werden. Nach der Speicherung der Konfigurationsdatei ist ein erneutes Stoppen und Starten von rvsEVO notwendig.

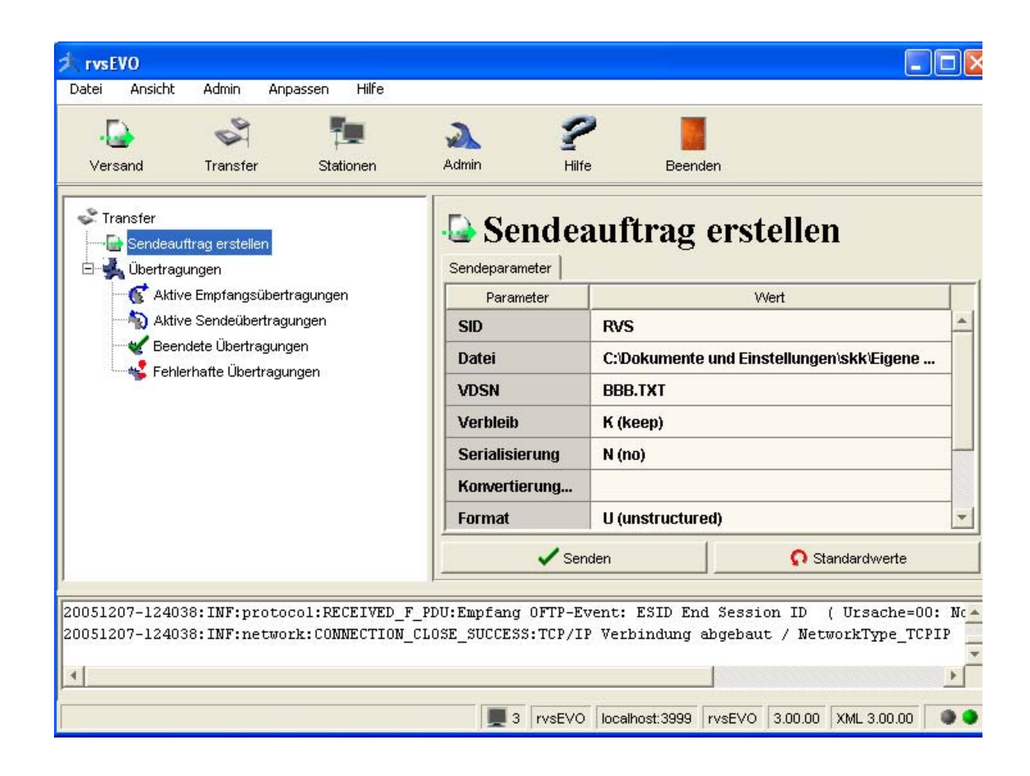
4.5 Versand einer Datei

`createSendJob`

Der Versand einer Datei ist mittels der GUI und mittels der Eingabeaufforderung möglich.

GUI

Wenn Sie das Symbol Versand oder Transfer in der Funktionsleiste anklicken, gelangen Sie ins Transfer-Fenster, wo Sie im Transfer-Baum den Menüpunkt **Sendeauftrag erstellen** wählen können. Im Fenster **Sendeauftrag erstellen** können Sie Ihre Daten für den aktuellen Sendeauftrag eintragen oder auswählen.



Pflicht-Parameter:

SID	StationsID des Empfängers
Datei	Dateiname der zu versendenden Datei
VDSN	Virtueller Dateiname; das ist der Dateiname, der für die ODETTE-Übertragung benutzt wird, darf maximal 26 Zeichen lang sein.

In der folgenden Tabelle werden die optionalen Parameter beschrieben:

Dateibeschreibung

Textkommentar, möglich erst ab der OFTP Version 2. Wenn Ihre Gegenstelle kein OFTP 2.0 kann, wird dieses Feld ignoriert.

Verbleib

Dieser Parameter entscheidet, ob die zu versendende Datei nach dem Versand bei Ihnen lokal gelöscht wird oder erhalten bleibt. Mögliche Werte:

– **K** (Keep): Datei bleibt nach dem Versand erhalten.

– **D** (Delete): Datei wird nach dem Versand gelöscht.

Serialisierung

Y(Yes)/**N**(No) Diese Option gewährleistet, dass Ihre Dateien in einer geordneten Reihenfolge ankommen. Alle Dateien, die in der gleichen Gruppe versendet werden sollen, müssen die gleiche Kennung (Label) haben. In der GUI dient VDSN als Label.

Sicherheitsmerkmale

Dieser Parameter bezieht sich auf die Art der Verschlüsselung. Je nachdem welche Art ausgewählt wird, werden die Parameter, die sich auf die jeweilige Verschlüsselungsart beziehen, aktiviert.

Folgende Werte sind möglich:

– 1 / ohne: keine Verschlüsselung ist erwünscht
– 2 / ComSecure (V1)
– 3 / ComSecure (V2)
– 4 / OFTP 2.0 (CMS)

rvsEVO kann in zwei verschiedenen Formaten verschlüsseln: ComSecure (ein eigenes Fomat) und CMS (basiert auf X.509-Zertifikaten).

CMS ist nur im Zusammenhang mit OFTP Version 2 möglich (auch die Gegenstelle muss OFTP Version 2.0 können).

Bei dem Format ComSecure wird zwischen der Version 1 (Produktversionen 1.1 und 1.2) und 2 (Produktversion 1.3) unterschieden. Version 2 wird verlangt, wenn Dateien größer als 4 Gbyte vor der Übertragung verschlüsselt werden sollen. Die Gegenstelle muss auch mit ComSecure Version 2 verschlüsseln können. Wenn es sich um kleinere Dateien handelt und die Gegenstelle das Format Com-Secure Version 1 verwendet, ist 2 /ComSecure (V1) ausreichend.

Offline Komprimierung

Odette-Komprimierung während der Übertragung. Mögliche Werte: **Y**(Yes)/**N**(No).
Standard: **N**

Verschlüsselung	Legt fest, ob Dateiverschlüsselung beim Dateitransfer angewandt wird. Mögliche Werte: **Y**(Yes)/**N**(No). Standard: **N** Mehr über Verschlüsselung lesen Sie bitte im Kapitel 6.
Verschlüsselungsalgorithmus	Dieser Parameter steht nur dann zur Verfügung, wenn als Sicherheitsmechanismus 4 /OFTP 2.0 (CMS) ausgewählt wurde. Folgende Algorithmen stehen zur Auswahl: – Kein – DES_EDE3_CBC (Triple DES) – AES256_CBC
Datei-Signatur	Dieser Parameter wird nur im Zusammenhang mit dem Sicherheitsmechanismus 4 /OFTP 2.0 (CMS) aktiviert. Mögliche Werte: – J (ja): Datei wird signiert. – N (nein): Datei wird nicht signiert.
signierten EERP/NERP (Quittungssignatur) beantragen	Nur im Zusammenhang mit dem Sicherheitsmechanismus 4 / OFTP 2.0 (CMS). Laut OFTP 2.0 haben Sie die Möglichkeit die Signatur der Quittung (EERP oder NERP) von Ihrem Partner zu beantragen. Allerdings muss der Partner auch die OFTP Version 2 unterstützen. Im Falle, dass Sie eine signierte Quittung beantragen und Ihr Partner (Destination - Endstation) sie Ihnen nicht sendet, endet der rvsEVO-Sendjob in den Status FAILED (fehlgeschlagen). Mögliche Werte: – J (ja) – N (nein).
Konvertierungstabelle	Für ASCII(ANSI) - EBCDIC Umwandlung stehen folgende Konvertierungstabellen zur Verfügung: ASCII-IBM037, ASCII-IBM237, ANSI-IBM037, ANSI-IBM273. Für EBCDIC - ASCII(ANSI) Umwandlung stehen folgende Konvertierungstabelle zur Verfügung: IBM037-ASCII, IBM237-ASCII, IBM037-ANSI, IBM273-ANSI

Format

Format der zu übertragenden Datei:

- T (Text): eine Folge von ASCII-Zeichen
- F (Fest): feste Satzlänge
- V (Variabel): variable Satzlänge
- U (Unstrukuriert): binäre Datei.

Die Dateien im Format F oder V müssen Textdateien mit der gewünschten Satzlänge (ohne CR/LF) sein.

Beispiel: Wenn Sie eine Datei im Format Fixed mit der Satzlänge 80 versenden möchten, muss diese Datei eine Textdatei sein, in welcher jede Zeile die Länge 80 ohne CR/LF hat. Dann sind die folgenden Parameter zu setzen: `Format=F`, `MaxRecl=80`.

Max.Satzlänge

Für Dateien im Format Fest geben Sie die feste Satzlänge an, mit der der Empfänger die Datei interpretieren soll. Dies ist die Satzlänge jedes Satzes bis zum Zeilenwechsel (CR/LF bei MS Windows, LF bei UNIX-Systemen). Für Dateien im Format Variabel geben Sie die maximale Satzlänge an. Siehe Beispiel für Parameter Format.

Umwandlungstabellen

Hinweis: Folgender Absatz bietet Ihnen eine kurze Beschreibung der verschiedenen Codeumwandlungstabellen, die von rvsEVO unterstützt werden:

- **ASCII**: US-ASCII ISO 646; Bei diesem Zeichensatz handelt sich um einen 7-Bit-Code, d.h. es sind maximal 128 Zeichen (Codewerte 0 - 127) darstellbar. Dieser Zeichensatz ist eine Teilmenge vieler anderer Zeichensätze mit 256 Zeichen, unter ihnen auch des ANSI - Zeichensatzes für MS Windows.
- **ANSI**: Windows ANSI; Die Zeichen an den Stellen 0 bis 127 sind gleich wie beim ASCII-Zeichensatz und die meisten Zeichen an den Stellen 128 bis 255 sind gleich wie beim ISO-Latin 1-Zeichensatz.
- **EBCDIC 037**: unterstützt Zeichen aus den folgenden Länder: Australien, Brasilien, Kanada, Neuseeland, Portugal, Südafrika, USA.
- **EBCDIC 273**: unterstützt Zeichen (insbesondere Umlaute) aus den folgenden Länder: Deutschland, Österreich, Schweiz.

Eingabeaufforderung

`createSendJob`

Mit dem Programm `createSendJob` erstellen Sie einen Sendeauftrag.

Syntax:

```
createSendJob -d <filename> -s <receiver sid>
-v <vdsn>
```

```
[-I <input code> -O <output code>]
[-t <table name>] [-F <format>]  [-M <length>]
[-j <start job>] [-S <serialize> -l <label>]
[-D <disposition>] [-C] [-Y]
[-h?] [-verbose]
```

Benötigte Parameter:

-d <filename> Dateiname der zu versendenden Datei.

-s <receiver sid> StationsID des Empfänger.

-v <vdsn> Virtueller Dateiname; das ist der Dateiname, der für die ODETTE-Übertragung benutzt wird, darf maximal 26 Zeichen lang sein.

Optionale Sende-Parameter:

-desc <description> Dateibeschreibung (nur

-F <format> Format der zu übertragenden Datei:

- T (Text): eine Folge von ASCII-Zeichen
- F (Fest): feste Satzlänge
- V (Variabel): variable Satzlänge
- U (Unstrukuriert): binäre Datei.

Die Dateien im Format F oder V müssen Textdateien mit der gewünschten Satzlänge (ohne CR/LF) sein.

Beispiel: Wenn Sie eine Datei im Format Fixed mit der Satzlänge 80 versenden möchten, muss diese Datei eine Textdatei sein, in welcher jede Zeile die Länge 80 ohne CR/LF hat. Dann sind die folgenden Parameter zu setzen: `-F=F`, `-M=80`.

-m <length> Für Dateien im Format Fest geben Sie die feste Satzlänge an, mit der der Empfänger die Datei interpretieren soll. Dies ist die Satzlänge jedes Satzes bis zum Zeilenwechsel (CR/LF bei MS Windows, LF bei UNIX-Systemen). Für Dateien im Format Variabel geben Sie die maximale Satzlänge an. Siehe Beispiel für Parameter Format.

-I <input code>	Für ASCII(ANSI) - EBCDIC Umwandlung stehen folgende Konvertierungstabellen zur Verfügung: ASCII-IBM037, ASCII-IBM237, ANSI-IBM037, ANSI-IBM273. Für EBCDIC - ASCII(ANSI) Umwandlung stehen folgende Konvertierungstabelle zur Verfügung: IBM037-ASCII, IBM237-ASCII, IBM037-ANSI, IBM273-ANSI. **Beispiel**: Wenn Sie die Umwandlungstabelle IBM237-ASCII benutzen möchten, geben Sie als Input Code IBM237 (`-I IBM237`) und als Output Code ASCII (`-O ASCII`) an.
-O <output code>	siehe Input Code.
-t <table name>	Pfad und Name Ihrer eigenen Codeumwandlungstabelle. Lesen Sie bitte das rvsXP-Benutzerhandbuch, Kapitel „Codeumwandlung“ für mehr Informationen.
-S <serialize>	**Y**(Yes)/**N**(No). Diese Option gewährleistet, dass Ihre Dateien in einer geordneten Reihenfolge ankommen. Alle Dateien, die in der gleichen Gruppe versendet werden sollen, müssen die gleiche Kennung (Label) haben. In der GUI dient VDSN als Label und hier (in der Eingabeaufforderung) kann Label (siehe Parameter Label) angegeben werden.
-l <label>	Alle Dateien, die in der gleichen Gruppe versendet werden, müssen die gleiche Kennung (Label) haben. Siehe auch Option `-s <serialize>`
-D <disposition>	Diese Option legt fest, ob die versendende Datei nach erfolgreichem Versand gelöscht werden soll. Mögliche Werte: **K** (keep) - Datei wird nicht gelöscht, Standard; **D** (delete) - Datei wird gelöscht.

-j <start job>	Diese Option legt fest, welches Skript gestartet werden soll, sobald der Sendeeintrag erzeugt wurde. Im Parameter `<start job>` soll der Name des Skriptes definiert werden. Der Skript selbst muss sich im Verzeichnis `$RVSTINY_HOME/bin` befinden. Über GUI ist dieser Parameter im Feld **JobAfterSE-Creation** zu setzen (`Admin Fenster -> Parameter`). Als Parameter wird diesem Skript die Transmission-ID übergeben.
-nocp	Beim Erzeugen eines Sendeeintrags wird standardmäßig die zu versendende Datei ins Oubox-Verzeichnis von rvsEVO kopiert und von hier versendet. Mit der Option `-nocp` kann dies unterbunden werden. In diesem Falle wird die Originaldatei nach erfolgreichem Versand gelöscht. **Achtung!** Diese Option hat jedoch keine Auswirkung, wenn die Datei im Format `U` mit Kodekonvertierung versendet wird!
-sfs <set-id>	Entspricht dem Parameter **Sicherheitsmerkmale** in der GUI. Nur mit **sfs=4** (OFTP 2.0). Mögliche Werte: – 1 (None) – 2 (ComSecure V1) – 3 (ComSecure V2 – 4 (OFTP 2.0)
-sif	Entspricht dem Parameter **Dateisignatur** in der GUI. Nur mit **sfs=4** (OFTP 2.0). Mögliche Werte: – J (Ja) – N (Nein)
-rsr	Entspricht dem Parameter **signierten EERP/ NERP beantragen** in der GUI. Nur mit **sfs=4** (OFTP 2.0). Mögliche Werte: – J (Ja) – N (Nein)

-desc <description>	Dateibeschreibung, möglich nur mit **sfs=3** (OFTP 2.0)
-C	ODETTE-Komprimierung (Offline-Komprimierung). Mögliche Werte: **Y**(Yes)/**N**(No). Standard: **N**
-Y	Verschlüsselung; Mögliche Werte: **Y**(Yes)/ **N**(No). Standard: **N**
-Yalg	Entspricht dem Parameter **Verschlüsselungsalgorithmus** in der GUI. Nur mit **sfs=3** (OFTP 2.0). Mögliche Werte: – 3DES – AES
-h	Fordert Hilfe (Usage) an.
-verbose	Ausgabe von ausführlichen Meldungen.
-?	Fordert Hilfe (Usage) an.

Beispiele:

```
createSendJob -d C:\text.txt -s RVS -v test
```

In diesem Beispiel wird die Datei `C:\text.txt` mit dem virtuellen Namen `test` an die Station `RVS` gesendet.

```
createSendJob -d C:\text.txt -s RVS -v OFTP_TEST
-F F -M 80 -I ANSI -O IBM273
```

In diesem Beispiel wird die Datei `C:\test.txt` an die Station `RVS` mit dem virtuellen Namen `OFTP_TEST` gesendet; diese Datei ist eine Textdatei, in welcher jede Zeile die Länge 80 (`-F F -M 80`) ohne CR/LF hat. Vor der Übertragung findet noch eine Codeumwandlung (ANSI-IBM273) statt.

```
createSendJob -d C:\part.txt -s RVS -v PART
 -S Y -l AUTO
```

In diesem Beispiel wird die Datei `C:\part.txt` an die Station `RVS` mit dem virtuellen Namen `PART` gesendet; diese Datei gehört zu der Gruppe der serialisierten Dateien mit dem Label `AUTO`.

Wenn ein Sendeauftrag erfolgreich generiert wurde, bekommen Sie die Meldung: `createSendJob exited with return code 0`.

Während der Sendeauftrag in Bearbeitung ist, entsteht im Verzeichnis `SND` eine temporäre Datei mit einem Namen wie z.B. `040329170027000`. Dieser Name setzt sich aus Datum (`040329`),

Uhrzeit (170027) und dreistelliger laufender Nummerierung (000) für Dateien, die in der gleichen Sekunde ankommen, zusammen.

Im folgenden Beispiel wartet der Job 040329170027000 auf eine Empfangsbestätigung (**EERP**) von der Station RVS und aus diesem Grunde befindet er sich noch im SND-Verzeichnis. SND und RCV sind Verzeichnisse, in denen die Jobs, die gerade bearbeitet werden, temporär abgelegt werden. Sie können im Feld **Status** lesen, im welcher Bearbeitungsphase sich der Job gerade befindet. WF_EERP (**W**aiting **F**or **E**ERP) bedeutet: warten auf EERP (Empfangsbestätigung).

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<Job>
       <ID>040329170027000</ID>
       <Static>
              <FileName>C:\Answer.txt</FileName>
              <VDSN>TESTDATEI</VDSN>
              <SID>RVS</SID>
              <Direction>SND</Direction>
              <Date>040329</Date>
              <Time>170027</Time>
       </Static>
       <Dynamic>
              <RestartPos>0</RestartPos>
              <RecCount>0</RecCount>
              <FilePos>19</FilePos>
                 <RcvBytes>0</RcvBytes>
                 <SendAttempts>0</SendAttempts>
                 <Status>WF_EERP</Status>
          </Dynamic>
</Job>
```

Wenn ein Job bis zum Ende erfolgreich abgearbeitet wurde (z.B. EERP ist angekommen), kopiert rvsEVO ihn ins Verzeichnis ENDED. Das Verzeichnis FAILED ist für die Jobs, die nicht erfolgreich abgearbeitet werden konnten.

Statuswerte

Einige mögliche Statuswerte für Sendjobs:

- RESTART=1 (Warte nach Fehler, um die Datei wieder zu senden)
- WF_SFID_ANSWER=2 (Warte auf SFID Antwort für bereits gesendete SFID)
- WF_CDT=3 (warte auf Credit)
- WF_EFID_ANSWER=4 (warte auf Antwort auf bereits gesendete EFID)
- WF_EERP=5 (Warte auf EERP)
- ENDED=6 (Sendjob ist beendet).
- ENDED_WITH_JS_ERROR=7 (Sendjob endete mit Fehler während des Aufrufs von Jobstart).
- FATAL_ERROR=8 (Schwerwiegender Fehler des Sendjobs)

Einige mögliche Statuswerte von Empfangjobs:

- RESTART=1 (Warte nach dem Fehler, um die Datei wieder zu empfangen).
- RESTART_AFTER_EFNA=2 (Die Datei konnte nicht empfangen werden, EFNA ist vielleicht gesendet, der Partner wird wahrscheinlich die Datei wieder senden).
- RESTART_AFTER_EFPA_FAILURE=3 (Die Datei wurde empfangen, aber EFPA war nicht erfolgreich. Der Partner wird wahrscheinlich die Datei erneut senden.)
- RECEIVING=4 (Empfang der Daten nach EFPA)
- EERP_HOLDED=5 (Die Datei wurde vollständig empfangen. EERP ist im HOLD-Status und soll freigegeben werden).
- EERP_RELEASED=6 (Der Benutzer hat EERP freigegeben, aber noch nicht versendet).
- EERP_DELETED=7 (Der Benutzer hat EERP gelöscht. Der Job wird gestoppt).
- ENDED=8 (Empfangsjob wurde erfolgreich beendet nach dem Versand von EERP).
- ENDED_WITH_JS_ERROR=9 (Receivejob endete mit Fehler während des Aufrufs von Jobstart).
- FATAL_ERROR=10 (Schwerwiegender Fehler des Empfangsjobs)

Hinweis: Lesen Sie bitte im rvs® portable, Referenzhandbuch (Kapitel 3.3) mehr zu dem ODETTE Protokoll (z.B. über EFNA, SFID, EFID, EERP,...).

4.6 Synchronisation von Sendeaufträgen

Einleitung

Der Dateitransfer über OFTP wird asynchron abgearbeitet. Bei Erstellung eines Sendeauftrags wird eine Datei lediglich zum Senden bereitgestellt. Der eigentliche Versand erfolgt, sobald rvs® eine Verbindung zur Partnerstation aufbauen konnte. Gerade im automatisierten Berieb ist es jedoch oft erforderlich, direkt zu reagieren, wenn eine Datei innerhalb einer bestimmten Zeit nicht versendet werden konnte, oder der Versand erfolgreich war.

rvsEVO stellt eine Methode zur Verfügung, mit der der Dateiversand quasi synchron abgewickelt werden kann. Diese Methode kehrt erst zum Anrufer zurück, wenn der Versand erfolgreich war oder ein Fehler auftrat. Es kann die Anzahl der Sendeversuche oder die Zeit angegeben werden, innerhalb der die erfolgreiche Abwicklung des Dateiversandes erwartet wird. Wird dieser Rahmen überschritten, gilt der Versand als fehlerhaft. Der Versand gilt als erfolgreich, wenn eine OFTP-Quittung (EERP) empfangen wurde.

Mit dieser Methode kann außerdem vor dem Dateiversand eine Konvertierung einer EDI Nachricht mit dem EDI-Konverter **WEDIConv** durchgeführt werden.

`convertAndSend`

Das Programm `convertAndSend` bietet Ihnen diese Funktionalität. Dieses Programm ist dem Programm `createSendJob` ähnlich. Es hat zusätzlich die Funktionalität zum synchronisierten Dateiversand und für die Konvertierung der EDI-Nachricht mit dem EDI-Konverter **WEDIConv**. Die Konvertierung kann in zwei Schritten erfolgen. Dem ersten Schritt kann hierbei ein XSLT-Stylesheet mitgegeben werden, mit dem das Ergebnis von dem XML-Format nochmals transformiert wird. Lesen Sie bitte das WEDIConv Benutzerhandbuch, um mehr Informationen über die Konvertierung einer EDI-Nachrichten zu erhalten.

Syntax:

```
convertAndSend -d <filename> -s <receiver sid>
-v <vdsn> [-I <input code> -O <output code>
-t <table name>] [-j <start job>] [-F <format>]
[-M <length>] [-S <serialize> -l <label>]
[-D <disposition>] [-nocp] [-za <attempts>]
[-zt <timeout>] [converter parms]  [-h?] [-verbose]
```

Benötigte Parameter:

-d <filename>	Dateiname der zu versendenden Datei.
-s <receiver sid>	StationsID des Empfängers.
-v <vdsn>	Virtueller Dateiname; das ist der Dateiname, der für die ODETTE-Übertragung benutzt wird, darf maximal 26 Zeichen lang sein.

Optionale Sende-Parameter:

-F <format>

Format der zu übertragenden Datei:

- `T` (Text): eine Folge von ASCII-Zeichen
- `F` (Fest): feste Satzlänge
- `V` (Variabel): variable Satzlänge
- `U` (Unstrukturiert): binäre Datei.

Die Dateien im Format `F` oder `V` müssen Textdateien mit der gewünschten Satzlänge (ohne CR/LF) sein.

Beispiel: Wenn Sie eine Datei im Format Fixed mit der Satzlänge 80 versenden möchten, muss diese Datei eine Textdatei sein, in welcher jede Zeile die Länge 80 ohne CR/LF hat. Dann sind die folgenden Parameter zu setzen: `-F=F`, `-M=80`.

-m <length>

Für Dateien im Format Fest geben Sie die feste Satzlänge an, mit der der Empfänger die Datei interpretieren soll. Dies ist die Satzlänge jedes Satzes bis zum Zeilenwechsel (CR/LF bei MS Windows, LF bei UNIX-Systemen).

Für Dateien im Format Variabel geben Sie die maximale Satzlänge an. Siehe Beispiel für Parameter Format.

-I <input code>

Für ASCII(ANSI) - EBCDIC Umwandlung stehen folgende Konvertierungstabellen zur Verfügung: ASCII-IBM037, ASCII-IBM237, ANSI-IBM037, ANSI-IBM273.

Für EBCDIC - ASCII(ANSI) Umwandlung stehen folgende Konvertierungstabellen zur Verfügung:
IBM037-ASCII, IBM237-ASCII, IBM037-ANSI, IBM273-ANSI.

Beispiel: Wenn Sie die Umwandlungstabelle IBM237-ASCII benutzen möchten, geben Sie als Input Code IBM237 (`-I IBM237`) und als Output Code ASCII (`-O ASCII`) an.

-O <output code>

siehe Input Code.

-t <conversion table>

Pfad und der Name Ihrer eigenen Codeumwandlungstabelle. Lesen Sie bitte das rvsXP Benutzerhandbuch, Kapitel „Codeumwandlung“ für mehr Informationen.

-j <start job>

Diese Option legt fest, welches Skript gestartet werden soll, sobald der Sendeeintrag erzeugt wurde. Im Parameter `<start job>` soll der Name des Skriptes definiert werden. Das Skript muss sich im Verzeichnis `$RVS_HOME/bin` befinden. Über die GUI ist dieser Parameter im Feld **JobAfterSECreation** zu setzen (`Admin Fenster -> Parameter`). Als Parameter wird diesem Skript die Transmission-ID übergeben.

-S <serialize>

Y(Yes)/**N**(No) Diese Option gewährleistet, dass Ihre Dateien in einer geordneten Reihenfolge ankommen. Alle Dateien, die in der gleichen Gruppe versendet werden sollen, müssen die gleiche Kennung (Label) haben. Siehe `-l <label>`.

-l <label>

Alle Dateien, die in der gleichen Gruppe versendet werden, müssen die gleiche Kennung (Label) haben. Siehe auch Option `-s <serialize>`.

-D <disposition>

Diese Option legt fest, ob die versendende Datei nach erfolgreichem Versand gelöscht werden soll. Mögliche Werte: **K** (keep) - Datei wird nicht gelöscht; **D** (delete) - Datei wird gelöscht; Standard bei `ConvertAndSend`.

-nocp

Beim Erzeugen eines Sendeeintrags wird standardmäßig die zu versendende Datei ins Oubox-Verzeichnis von rvsEVO kopiert und von hier versendet. Mit der Option `-nocp` kann dies unterbunden werden. In diesem Falle wird die Originaldatei nach erfolgreichem Versand gelöscht. **Achtung!** Diese Option hat jedoch keine Auswirkung, wenn die Datei im Format `U` mit Kodekonvertierung versendet wird!

-za <attempts>

Anzahl von Sendeversuchen für synchronisierte Übertragung.

-zt <timeout>

Timeout in Sekunden für synchronisierte Übertragung.

-C

ODETTE-Komprimierung (Offline-Komprimierung). Mögliche Werte: **Y**(Yes)/**N**(No).
Standard: **N**

-Y

Verschlüsselung; Mögliche Werte: **Y**(Yes)/**N**(No).
Standard: **N**

Umwandlungs tabellen

Hinweis: Folgender Absatz bietet Ihnen eine kurze Beschreibung der verschiedenen Codeumwandlungstabellen, die von rvsEVO unterstützt werden:

- **ASCII**: US-ASCII ISO 646; Bei diesem Zeichensatz handelt sich um einen 7-Bit-Code, d.h. es sind maximal 128 Zeichen (Codewerte 0 - 127) darstellbar. Dieser Zeichensatz ist eine Teilmenge vieler anderer Zeichensätze mit 256 Zeichen, unter ihnen auch des ANSI-Zeichensatzes für MS Windows.
- **ANSI**: Windows ANSI; Die Zeichen an den Stellen 0 bis 127 sind gleich wie beim ASCII -Zeichensatz und die meisten Zeichen an den Stellen 128 bis 255 sind gleich wie beim ISO-Latin 1-Zeichensatz.
- **EBCDIC 037**: unterstützt Zeichen aus den folgenden Länder: Australien, Brasilien, Kanada, Neuseeland, Portugal, Südafrika, USA.
- **EBCDIC 273**: unterstützt Zeichen (insbesondere Umlaute) aus den folgenden Länder: Deutschland, Österreich, Schweiz.

Optionale EDI-Konvertierungsparameter:

-cf <Dateipfad<	Format-Beschreibung für den ersten Schritt des EDI-Konverters.
-cf2 <Dateipfad<	Format-Beschreibung für den zweiten Schritt des EDI-Konverters (optional).
-ct <Dateipfad>	Stylesheet für den ersten Schritt des EDI-Konverters.
-cd	EDI-Konverter-Richtung für den ersten Schritt, default EDI2XML.
-cd2	EDI-Konverter-Richtung für den zweiten Schritt, default XML2EDI.
-cl <Ganzzahl>	log level für den EDI-Konverter, default:0
-cs <0|1>	Verhalten am Zeilenende, für die EDI-Nachricht-Ausgabe: 0 - kein LF als Segment-Trenner 1 - mit LF als Segment-Trenner.
-cI <0|1>	Einrückung für XML-Ausgabe: `0` - ohne Einrückung; `1` - mit Einrückung (default).
-ce <encoding>	Encoding für die XML-Ausgabe; default UTF-8.

Optionale Parameter:

-verbose	Ausgabe von ausführlichen Meldungen.

-help	Gibt Beschreibung zum aktuellen Kommando aus
-?	Fordert Hilfe an.

Beispiele:

```
convertAndSend -d C:\teil56.txt -s RVS -v TEILE
-za 4
```

In diesem Beispiel wird die Datei `C:\teil56.txt` zur Station `RVS` mit dem virtuellen Dateinamen `TEILE` gesendet; die Anzahl der Sendeversuche ist begrenzt auf 4.

```
convertAndSend
-d C:\INTEGRATION\test.txt -s RVS -v TEST
-cf C:\rvsET\system\fmtDesc\fw.kanban.ineas.xml
-ct C:\rvsET\system\stylesheets\ineas2deljit.xslt
-cf2 C:\rvsET\system\fmtDesc\edifact.97.orig.xml
```

EDI-Konverter

In diesem Beispiel wird die Datei `C:\INTEGRATION\test.txt` zur Station `RVS` mit dem virtuellen Dateinamen `TEST` gesendet. Die Datei wurde auch mit dem EDI-Konverter **WEDIConv** (der in dem Verzeichnis `C:\rvsET` installiert wurde) konvertiert, zuerst vom Inhouse-Format (Datei `test.txt`) zum XML-Format (Datei `fw.kanban.ineas.xml`) und dann zur EDIFACT-Nachricht (Datei `edifact.97.orig.xml`). Beim ersten Schritt wurde für die Darstellung folgendes Stylesheet benutzt: `ineas2deljit.xslt`.

Beachten!

Hinweis: Bei der Benutzung dieses Programms ist zu beachten, dass ein erneuter Sendeauftrag nach Fehler doppelte Übertragungen zur Folge haben kann. Wenn z.B. die Datei schon zur direkten Gegenstelle übertragen werden konnte, jedoch innerhalb der vorgegebenen Übertragungszeit die Quittung nicht eintraf, ist die Übertragung unabhängig von der eigenen Versandstation weiterhin aktiv. Es kann sein, dass die Datei von unserer direkten Partnerstation trotzdem noch übertragen wird.

Dateizähler

Darauf hat die eigene Station keinen Einfluss mehr. Ein erneutes Versenden der Datei hat dann die doppelte Übertagung zur Folge. Die Anwendung in der Schicht über rvs® muss also mit einer solchen Situation umgehen können. In der Regel wird dies gelöst, indem die Datei anhand des Dateinamens eindeutig ist, z.B. durch Verwendung eines aufsteigenden Zählers oder indem die Anwendung die Eindeutigkeit über den Dateiinhalt feststellt.

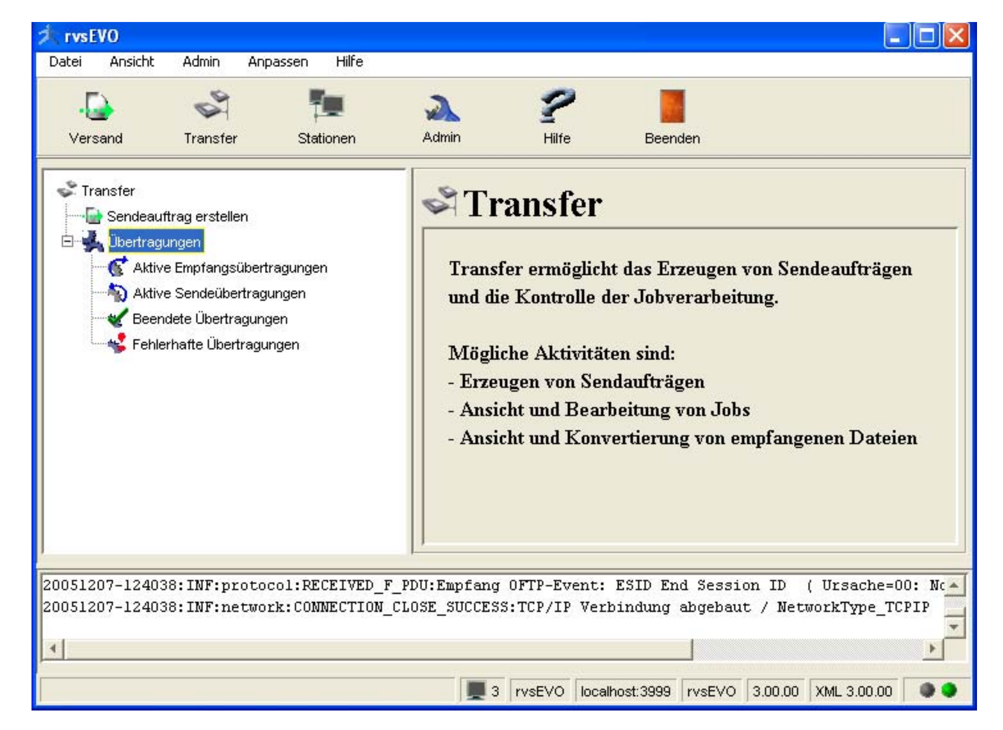
4.7 Ausgabe aller Empfangs- und Sendejobs

Die wichtigsten Informationen über die Sende- und Empfangsjobs können über die GUI oder mit dem Program `getJobList` angezeigt werden. Wie Sie die rvsEVO GUI starten können, lesen Sie bitte im Kapitel .

GUI

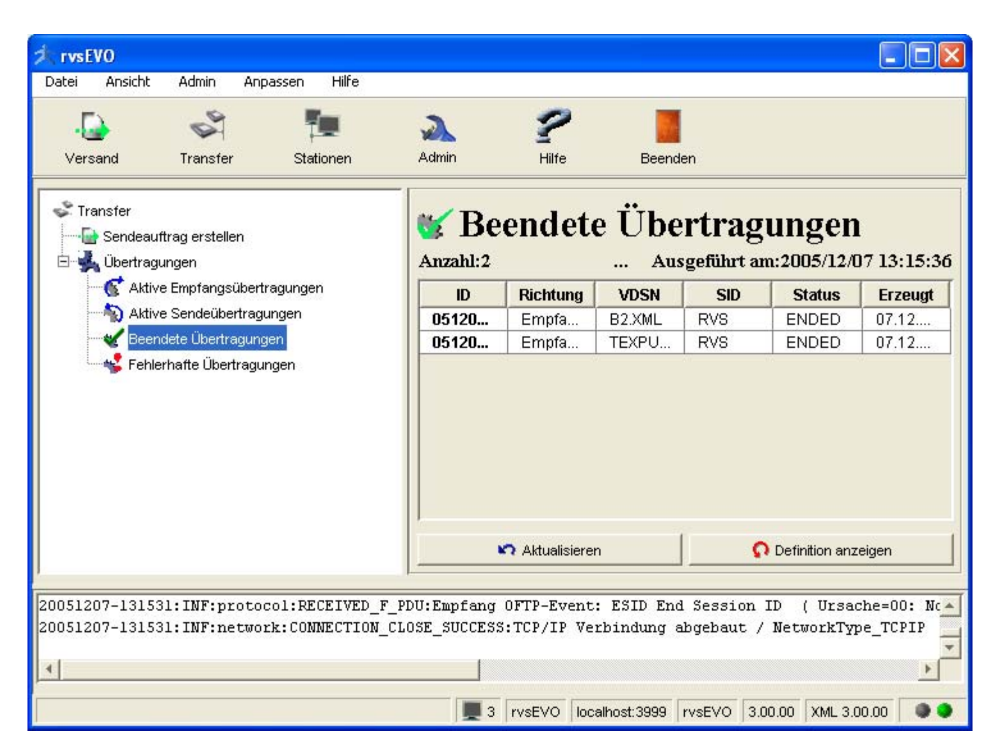
`Übertragungen`

Wenn Sie das Symbol Transfer in der Funktionsleiste wählen, öffnet sich das Fenster mit dem Ordner `Übertragungen` in dem Transferbaum. Sie können jetzt zwischen Aktiven Empfangs/Sendeübertragungen, Beendeten Übertragungen (Sende- und Empfangs-Richtung) und Fehlerhaften Übertragungen (Sende- und Empfangs-Richtung) wählen.



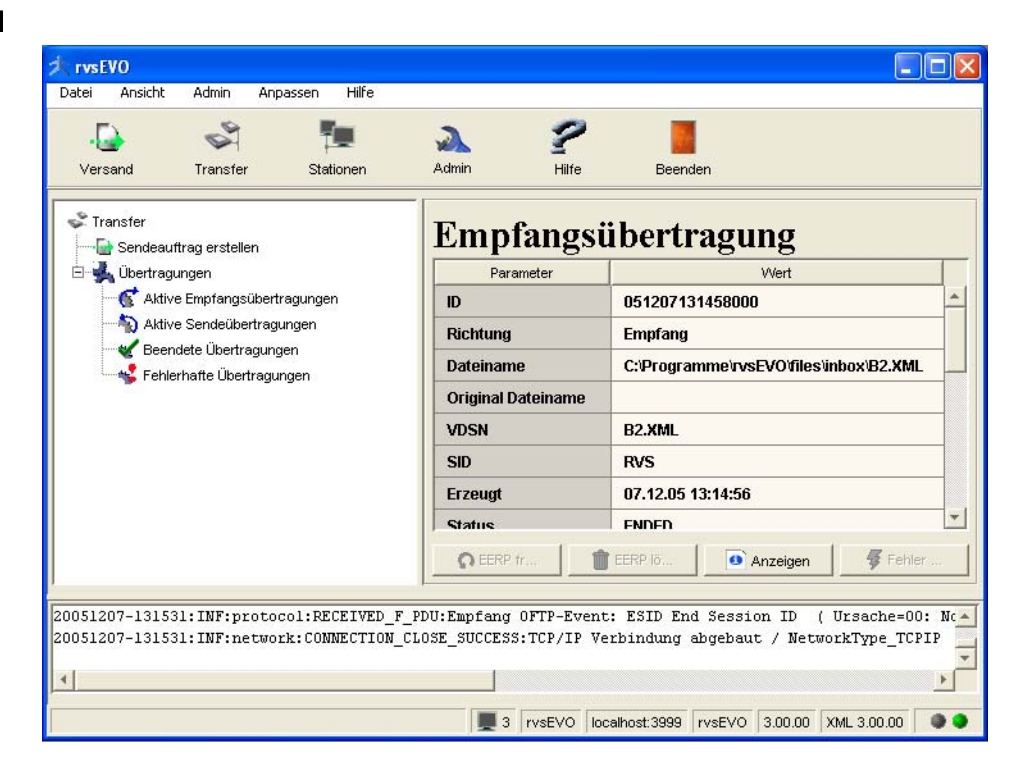
Hinweis: Eine Übertragung gilt erst als abgeschlossen, wenn die ODETTE-Quittung EERP (End-to-End-Response) für diese Übertragung angekommen ist.

Eine Übersicht aller vorhandenen Übertragungen bekommen Sie, indem Sie den gewünschten Übertragungs-Ordner mit einem Klick anwählen (markieren).



Die Daten eines Jobs können Sie sich anzeigen lassen, in dem Sie mit der Maus einen Doppelklick auf eine Jobzeile im rechten Teil des Fensters ausführen.

Beispiel



EERP

Bei den aktiven Empfangsübertragungen bietet rvsEVO die Möglichkeit den EERP anzuhalten, freizugeben oder zu löschen. Einen EERP anhalten können Sie, indem Sie im rechten Fensterteil die Schaltfläche **Anhalten** betätigen; wieder freigeben, indem Sie die Schaltfläche **Freigeben** und löschen, indem Sie die Schaltfläche **Entfernen**, betätigen. Je nachdem für welche Aktion Sie sich entscheiden, ändert sich der Wert in der Zeile **Status** (z.B. `EERP_HOLDED`). Das Gleiche können Sie auch mit dem Programm `handleEERP` erzielen (siehe Kapitel 4.9).

Bei den aktiven Sendeübertragungen haben die gleichen Schaltflächen im unteren Teil des einzelnen Jobfensters (Detailansicht) die Aufgabe, einen Sendejob anzuhalten, zu löschen oder wieder freizugeben. Ein aktiver Sendejob muss zuerst angehalten werden, um gelöscht werden zu können.

Eingabeaufforderung

`getJobList`

Mit dem Programm `getJobList` können Sie alle Jobs auflisten.

Syntax:

```
getJobList  [-a] [-e] [-f] [-verbose]
```

Optionale Parameter:

-a	Ausführliche Ausgabe der Informationen über Jobs, die in Bearbeitung sind.
-e	Ausgabe der Informationen über beendete Jobs.
-f	Ausgabe der Informationen über fehlgeschlagene Jobs.
-h	Fordert Hilfe (Usage) an.
-verbose	Ausgabe von ausführlichen Meldungen.
-?	Fordert Hilfe (Usage) an.

Beispiel:

```
getJobList -e
```

Ergebnis:

```
C:\Programme\rvsEVO\bin>getJobList -e
***********************
job 051207131458000 (RCV): state=ENDED
job 051207131530000 (RCV): state=ENDED
***********************
List ended. size=2
***********************
C:\Programme\rvsEVO\bin>
```

4.8 Ausgabe eines JobEintrages

`getJob` Mit dem Programm `getJob` erhalten Sie Informationen zu einem bestimmten Sende- oder Empfangsjob.

Syntax:

```
getJob -n <jobid> [-a] [-verbose]
```

Benötigte Parameter:

-n <jobid>	Angabe eines Sende- oder Empfangsjobs mit der ID `<jobid>`.

Optionale Parameter:

-a	Ausgabe der komplett verfügbaren Job-Informationen.
-help	Fordert Hilfe (Usage) an.
-verbose	Ausgabe von ausführlichen Meldungen.
-?	Fordert Hilfe (Usage) an.

Beispiel:

```
getJob -n 040329175603000
```

Ergebnis: `job 040329175603000 (SND): state: WF_EERP`

4.9 Löschen oder Freigeben von EERPs

`handleEERP`

Mit dem Programm `handleEERP` können Sie Empfangbestätigungen (`EERP_OUT`) löschen oder freigeben.

Syntax:

```
handleEERP -r|-d <JobID> [-verbose]
```

Benötigte Parameter:

-r | -d <JobID>	ID des Jobs für welchen die EERP freigegeben / gelöscht werden soll.

Optionale Parameter:

-help	Fordert Hilfe (Usage) an.
-verbose	Ausgabe von ausführlichen Meldungen.
-?	Fordert Hilfe (Usage) an.

4.10 Archivieren der Einträge der abgearbeiteten Sende- bzw. Empfangsaufträge im Revisionslog

`archiveJobs`

Mit dem Programm `archiveJobs` wird im Verzeichnis `$RVS_HOME/rvsEVO/archive` eine Datei `RevisionLog.xml` erzeugt, in der Einträge der erfolgreich abgearbeiteten Sende- bzw. Empfangsaufträge gespeichert sind.

Syntax:

```
archiveJobs [-d <date>] [-help] [-?]
```

Optionale Parameter:

-d <date>	Dieser Parameter definiert ein Datum in der Form `yyMMddHHmmss` oder `yyMMdd` mit dem festgelegt wird, dass nur Übertragungen archiviert werden, die älter als angegebenes Datum sind.
-help	Fordert Hilfe (Usage) an
-?	Fordert Hilfe (Usage) an.

Dabei werden beendete Jobs aus dem Verzeichnis `$RVS_HOME\jobs\ENDED` gelöscht und die Daten in die Datei `RevisionLog.xml` im Verzeichnis `$RVS_HOME\archive` geschrieben.

Hinweis: Mehrere `RevisionLog.xml`-Dateien werden durch einen Zeitstempel unterschieden.

5 Sicherung und Wiederherstellung der rvsEVO-Daten

rvsEVO bietet die Möglichkeit, eine Sicherung aller relevanten Daten durchzuführen und bei Bedarf diese Sicherung wieder einzuspielen. Dieses Verfahren ist inbesondere dann von Bedeutung, wenn ein Fehler in rvsEVO aufgetreten ist und der Benutzer den alten Zustand vor dem Fehler wiederherstellen möchte.

5.1 Sicherung (Backup)

Voraussetzung: Die Funktion Sicherung wird erst dann gestartet, nachdem gewährleistet wurde, dass keine Kommunikation mehr stattfindet (kein Empfang/Senden der Dateien und keine Verschlüsselung/Komprimierung). Wenn die Prozesse noch laufen, werden sie von dem Sicherungs-Skript beendet.

Eine Sicherung kann in der rvsEVO GUI mit dem Menüpunkt `Admin/Backup` durchgeführt werden.

Alternativ kann diese Funktion durch Aufruf eines Skriptes auf der Kommandozeile gestartet werden.

Syntax:

```
createBackup [-d <dir>] [-verbose] [-help] [-?]
```

Alle Parameter sind optional.

-d <dir>	Angabe des Sicherungsverzeichisses; ohne diese Option werden die Sicherungsdaten ins Verzeichnis `$RVS_HOME/archive` geschrieben.
-verbose	Ausgabe von ausführlichen Meldungen.
-help	Fordert Hilfe (Usage) an.
-?	Fordert Hilfe (Usage) an.

Hinweis: Mit dem neuen Parameter `BackupOnStartup` kann konfiguriert werden, dass bei jedem Start von rvsEVO eine automatische Sicherung durchgeführt wird. Dieser Parameter ist über die grafische Oberfläche (Admin-Fenster) oder in der Konfigurationsdatei `$RVS_HOME/conf/rvsConfig.xml` zu setzen. Standardwert ist `Y` (Yes).

5.1.1 Was wird gesichert?

Folgende Dateien oder Verzeichnisse werden gesichert:

– Das ganze Verzeichnis `$RVS_HOME/conf`, welches die Konfigurationsdateien von rvsEVO beinhaltet.

– Sicherung der Jobdateien aus den Verzeichnissen `$RVS_HOME/jobs/SND` und `$RVS_HOME/jobs/RCV`. Hiermit werden die temporären, nicht vollständig abgearbeiteten Jobs gespeichert.

– Dateien aus dem Verzeichnis $RVS_HOME/system/data.

Aus den gesicherten Daten wird eine <BackupZeitpunkt>.jar-Datei erstellt. Mit BackupZeitpunkt ist der Dateiname gemeint, der automatisch bei der Sicherung vergeben wird und der Form YYMMDDHHMMSS mit einem zusätzlichen 3-stelligen Zähler besitzt.

Beispiel: Datei 051010112417000.jar ist die Sicherungsdatei, die am 10.10.05 um 11:24:17 erstellt wurde. Es wurden bisher keine anderen Dateien zu diesem Zeitpunkt erstellt und daher lautet der zusätzliche Zähler 000.

5.1.2 Redo-Log

Ab dem Zeitpunkt der Sicherung werden alle dynamischen Daten in einem fortlaufenden Log (Redo-Log) protokolliert.

Mit „dynamischen Daten“ sind die Informationen über die Sende- und Empfangsjobs gemeint. Diese Daten werden geschrieben, um die nicht vollendeten Übertragungsjobs später wiederherstellen zu können. Die vollendeten Jobs werden protokolliert, sind aber nicht von Bedeutung für die Wiederherstellung (Siehe 5.2).

Damit das Redo-Log eindeutig einer Sicherung zugeordnet werden kann, erhält es den gleichen Namen (Zeitstempel) wie die zugehörige Sicherungsdatei allerdings mit dem Präfix Redo_ und der Erweiterung .log. Diese Datei wird genauso wie die Sicherungsdatei, in das Verzeichnis $RVS_HOME/archive geschrieben.

Beispiel:

Zu der Sicherungsdatei 051010112417000.jar wurde die zugehörige Redo-Logdatei Redo_051010112418000.log erzeugt.

Sobald eine neue Sicherung erzeugt wurde, wird auch eine zugehörige Redo-Logdatei angelegt. Dabei wird bei allen vorhandenen Redo-Logdateien die Dateierweiterung .log durch .old ersetzt.

Die Redo-Logdatei ist eine Textdatei. Jede Zeile enthält ein XML-Element <Job> mit allen zugehörigen Jobinformationen wie z.B. ID, FileName, VDSN, SID.

5.2 Wiederherstellung der rvsEVO-Daten (Recovery)

Eine Wiederherstellung der rvsEVO-Daten erfolgt über die grafische Oberfläche, Menüpunkt Admin/Recover. In dem Dialog, welcher dann angeboten wird, kann man den Namen der Sicherungsdatei und des Redo-Logs angeben.

Wenn die Angabe des Dateinamens ohne Pfad erfolgt, werden die Sicherungsdatei und das Redo-Log im Verzeichnis $RVS_HOME/archive gesucht.

Alternativ kann diese Funktion auf der Kommandozeile mit dem Skript $RVS_HOME/bin/doRecover.bat gestartet werden.

Syntax:

```
doRecover -b <dir> -r <name>
[-help][-?]
```

-b <dir>	Pflichtparameter: Name der Sicherungsdatei; kann als kompletter Dateiname inklusive Verzeichnis oder als Dateiname ohne Verzeichnis angegeben werden. Im letzteren Fall wird die Sicherungsdatei im Verzeichnis `$RVS_HOME/archive` erwartet.
-r	Name der Redo-Logdatei. Er kann als kompletter Dateiname inklusive Verzeichnis oder als Dateiname ohne Verzeichnis angegeben werden. Im letzteren Fall wird die Redo-Log-Datei im Verzeichnis `$RVS_HOME/archive` erwartet.
-verbose	Ausgabe von ausführlichen Meldungen.
-help	Fordert Hilfe (Usage) an.
-?	Fordert Hilfe (Usage) an.

Hinweis: Vor dem Ausführen des Wiederherstellungsprozesses werden sowohl alle Sende- und Empfangsprozesse als auch der Service Provider (Verschlüsselung/Komprimierung) beendet.

Das Einspielen des Redo-Logs wird in der Datei `$RVS_HOME/log/monlog.log` protokolliert.

6 Verschlüsselte Übertragung mit rvsEVO

In diesem Kapitel werden die Grundlagen der Dateiverschlüsselung mit zwei Verschlüsselungsformaten (Com-Secure und CMS) und die Verwaltung der Schlüssel für den sicheren Dateiaustausch in rvsEVO beschrieben.

In rvsEVO ab der OFTP Version 2 wird für die Verschlüsselung auf der Session-Ebene das Protokoll TLS (Transport Layer Security) eingesetzt. Lesen Sie bitte das Kapitel 3.2 "Anpassen der Stationskonfiguration", um mehr darüber zu erfahren, wie Sie rvsEVO für TLS konfigurieren sollen. Das Prinzip der Schlüsselverwaltung (öffentliche und privater Schlüssel) ist das Gleiche für Com-Secure, CMS und für TLS.

6.1 Einleitung: Grundlagen

rvsEVO bietet zwei Formate für die Dateiverschlüsselung an: Com-Secure und CMS.

Com-Secure ist ein eigenes Format, welches auch für die Verschlüsselung in rvs® portable verwendet wird.

CMS (Cryptographic Message Syntax) ist ein Internet-Standard (rfc 2630), welcher den Aufbau einer verschlüsselten Nachricht beschreibt. Dieser Standard wird erst ab OFTP Version 2 unterstützt.

In Com-Secure und CMS werden die Vorzüge des symmetrischen und des unsymmetrischen Verfahrens kombiniert: die hohe Geschwindigkeit des symmetrischen Verfahrens und das Sicherheitsniveau des unsymmetrischen Verfahrens. rvsEVO setzt folgende Verfahren ein:

– **3DES** als symmetrisches Verfahren für ComSecure (Länge 3x56 Bit = 168 Bit)
– **3DES** und **AES** als symmetrische Verfahren für CMS
– **RSA** als unsymmetrisches Verfahren für ComSecure und CMS (Länge 768 bis 2048 Bit).

Elektronische Signatur

Zur Erhöhung der Sicherheit arbeitet die Verschlüsselungskomponente mit elektronischer Signatur. Mit der Signatur stellt man sicher, dass die Daten bei der Übertragung keine unbemerkte Veränderung erfahren.

Ab der OFTP Version 2 erstreckt sich die elektronische Signatur auch auf den End-To-End-Response (EERP) und den Negative-End-to-End-Response (NERP).

6.2 Prinzip und Ablauf der rvsEVO-Verschlüsselung

Die Prinzipien der Verschlüsselung, die im nächsten Abschnitt erläutert werden, sind grundsätzlich die gleichen für die beiden Verschlüsselungsformate (CMS und ComSecure).

Jeder Teilnehmer an der verschlüsselten Kommunikation erstellt auf seinem System ein Schlüsselpaar, bestehend aus dem **öffentlichen Schlüssel** und dem **privaten Schlüssel**.

Öffentlichen Schlüssel verteilen / privaten Schlüssel sicher aufbewahren

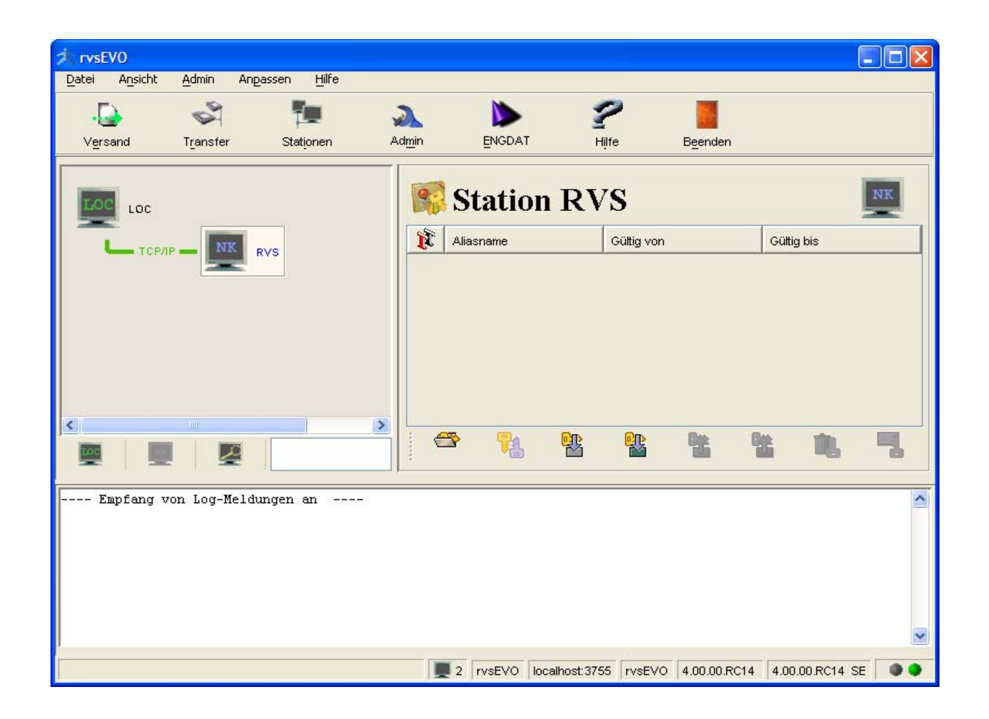
Den öffentlichen Schlüssel stellt er jedem Partner zur Verfügung, von dem er Dateien erwartet. Der jeweilige Sender kann damit die Daten genau für den Partner verschlüsseln, von dem dieser öffentliche Schlüssel stammt. Da der öffentliche Schlüssel allein zur Entschlüsselung nicht geeignet ist, kann man ihn ohne Sicherheitsrisiko offen übermitteln.

Den privaten Schlüssel behält jeder Teilnehmer nur für sich und bewahrt ihn sicher auf.

Für die Entschlüsselung werden drei Schlüssel benötigt (das eigene Schlüsselpaar und der öffentliche Schlüssel des Partners). Geht einer der drei benötigten Schlüssel verloren, dann ist es nicht mehr möglich, die vom Partner verschlüsselten Dateien zu entschlüsseln.

In den nächsten Unterkapiteln wird die Schlüsselverwaltung für die Dateiverschlüsselung beschrieben.

Zur Schlüsselverwaltung gelangen Sie, indem Sie die Schaltfläche im rechten oberen Teil des Stationsfensters betätigen.



Folgende Funktionalitäten stehen zur Verfügung:

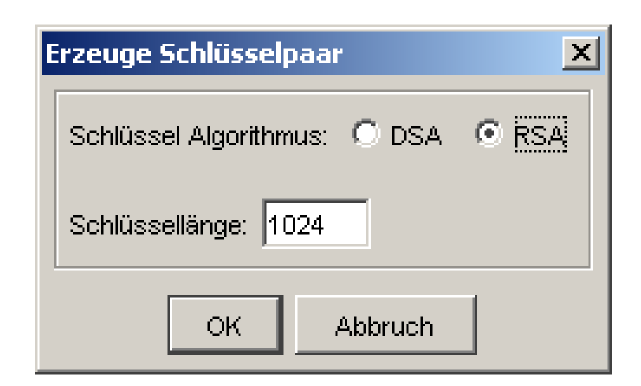
– Erzeuge die Liste der Schlüssel und Zertifikate
– Schlüsselpaar erzeugen
– Zertifikat importieren

– Öffentlichen Schlüssel für ComSecure importieren
– Zertifikat exportieren
– Zertifikat zu ComSecure exportieren
– Schlüssel löschen
– Erzeuge Zertifikatsanfrage

6.3 Wie erzeuge ich ein eigenes Schlüsselpaar?

Im unteren Teil des Schlüsselverwaltungsfensters finden Sie die Schaltfläche zum Erzeugen eines neuen Schlüsselpaars.

Im Fenster **Erzeuge Schlüsselpaar** soll als Schlüssel-Algorithmus (`Key Algorithm`) `RSA` und als Schlüsselgröße (`Key Size`) die Standardeinstellung 1024 gewählt werden.



Im nächsten Dialogfenster **Erzeuge Schlüsselpaar** soll als Signatur-Algorithmus (`Signature Algorithm`) `SHA1withRSA` gewählt werden. Mit Dauer (`Validity`) ist die Gültigkeitsdauer der Schlüssel in Tagen gemeint.

Der Parameter `Common Name` ist z.B. von Bedeutung, wenn es sich um eine Anbindung an eine vorhandene PKI (Public Key Infrastructure) handelt und diese Angabe für die LDAP-Struktur benötigt wird. Dieser Parameter ist obligatorisch.

Die restlichen Angaben betreffen Ihre Organisation. rvsEVO macht hier keine Vorschriften, wie diese Felder auszufüllen sind.

Nachdem Sie alle Angaben im Fenster **Erzeuge Schlüsselpaar** ausgefüllt und mit **OK** bestätigt haben, das Schlüsselpaar wird erzeugt und im rechten Fenster erscheint eine Zeile mit der Angabe der Gültigkeitsdauer des Schlüsselpaars.

Die Verwaltung der eigenen (privaten und öffentlichen) und der Partner-Schlüssel (nur öffentlichen) erfolgt in einer Schlüsselverwaltungsdatei (keystore).

Der Pfad der zu verwendenden Schlüsselverwaltungsdatei ist über die GUI oder in der Konfigurationsdatei `$RVS_HOME/conf/cryptoParameter.xml` (XML-Element: `keyStoreParameter`; Unterelement: `fileName`) zu setzen. Standardwert ist: `$RVS_HOME/system/data/keystore.p12`.

Hinweis: Die Endung `.p12` bedeutet, dass diese Schlüsselverwaltungsdatei im Format PKCS #12 ist.

Wie Sie eine verschlüsselte Datei mit rvsEVO versenden können, lesen Sie bitte im Kapitel 4.5 "Versand einer Datei".

6.4 Zertifikat importieren und exportieren

rvsEVO bietet die Möglichkeit vertrauenswürdige Zertifikate von Partnern in die Schlüsselverwaltung zu importieren und schon importierte Zertifikate aus der Schlüsselverwaltung in eine Datei zu exportieren.

Um ein X.509-Zertifikat in die rvsEVO-Schlüsselverwaltung zu importieren, klicken Sie auf die Schaltfläche **Schlüssel importieren**.

Das Fenster **Vertrauenswürdiges Zertifikat importieren** öffnet sich und Sie können die Datei auswählen, die importiert werden soll. Nachdem Sie **OK** gedrückt haben, ist das Zertifikat im Fenster Schlüselverwaltung sichtbar.

Umgekehrt ist es auch möglich, ein schon importiertes Zertifikat, als eine Zertifikatsdatei zu exportieren. Wählen Sie für diese Zwecke die Schaltfläche **Zertifikat exportieren**.

6.5 ComSecure-Schlüssel importieren und exportieren

Da rvsEVO auch ein eigenes Format für öffentliche Schlüssel unterstützt, ist es notwendig, beim Import/Export die Konvertierung aus/in ein X.509-Zertifikat (Standard-Format für digitale Zertifikate; Datenamenserweiterung .cer) vorzunehmen.

Es gibt zwei Schaltflächen, die diese Funktionalität unterstützen. Diese sind: **Öffentlichen Schlüssel für ComSecure importieren** und **Zertifikat zu ComSecure exportieren**.

Mit der ersten Funktionalität **Öffentlichen Schlüssel für ComSecure importieren** ist es möglich, einen öffentlichen Schlüssel im ComSecure-Format als ein X.509-Zertifikat in die rvsEVO-Schlüsselverwaltung zu importieren.

Die Funktionalität **Zertifikat zu ComSecure exportieren** ermöglicht ein in rvsEVO schon vorhandenes Zertifikat als ein öffentlicher Schlüssel im ComSecure-Format zu exportieren.

6.6 Restliche Funktionen

Die rvsEVO-Schlüsselverwaltung bietet noch zwei zusätzliche Funktionen.

Mit der Schaltfläche **Schlüssel löschen** wird ein Schlüsselpaar aus der Schlüsselverwaltung gelöscht.

Die Funktion **Erzeuge Zertifikatsanfrage** ermöglicht es Ihnen eine Zertifikatsanfrage (Zertifikatsrequest) im sog. Base64-Format zu erzeugen, um einen Antrag auf ein Serverzertifikat bei einem TrustCenter zu stellen.

7 rvsEVO-ENGDAT

In diesem Kapitel wird das rvsEVO-Modul ENGDAT vorgestellt. Dieses Modul dient zum Empfangen und Versenden von ENGDAT-Nachrichten. Nach einer allgemeinen Einführung in die ENGDAT-Norm werden die Funktionen des ENGDAT-Moduls in rvsEVO erklärt.

7.1 Einführung

Das ENGDAT-Modul in rvsEVO ist eine Spezialisierung von rvsEVO für die Übertragung von CAD/CAM-Informationen. Dieses Modul unterstützt ENGDAT-Versionen 1 und 2, sowie ENGDAT Version 3 im Umfang der Conformance Class 2 (CC2).

Die Versionen 1 und 2 stützten sich auf die europäisch genormte ODETTE-ENGDAT-Empfehlung (VDA-Empfehlung 4951). Diese Empfehlung berücksichtigt die in DIN ISO 9735 genormte EDIFACT-Syntax.

ENGDAT Version 3 wurde von der Projektgruppe SASIG (Strategic Automotive Product Data Standards Industry Group) entwickelt. Die größte Neuerung in der Version 3 ist die Umstellung auf XML als Nachrichtenstandard.

rvsEVO-ENGDAT arbeitet mit ENGDAT-Nachrichten (**Eng**ineering **Dat**a Message). ENGDAT-Nachrichten sind elektronische Lieferscheine für **Eng**ineering **Dat**en. Eine ENGDAT-Nachricht ist quasi ein Paket bestehend aus verschiedenen Dateien. Dies sind zum einen die Nutzdaten (CAD/CAM-Modelle, Stücklisten, Zusammenbaubeschreibungen, etc.) und zum anderen Metadaten (Informationen über Sender und Empfänger wie z.B. Telefonnummer, Firma, Abteilung, sowie Informationen über jede einzelne CAD/CAM-Datei aus dem jeweiligen ENGDAT-Paket wie z.B. Datentyp, Version, Kompressionsart). Die Nutzdaten werden über Metadaten (die Abstract-Datei) zusammengehalten, zu einer Nachricht zusammengefasst und versendet. Dies bedeutet, dass die Dateien zwar einzeln versendet werden, aber über die Abstract-Datei als ENGDAT-Paket identifiziert werden.

Die Abstract-Datei und die zugehörigen Nutz-Dateien haben bei der Datenübertragung einen speziellen 26-stelligen Dateinamen. Dieser Dateiname wird aus folgenden Komponenten zusammengesetzt:

ENG{Zeitstempel}{Freie Kennung}{Dateianzahl}{Sequenznummer}.

Es existieren gewisse Unterschiede bei der Dateinamenskonvention zwischen den Versionen 1 bis 2 (ENGDAT V1/V2) und der Version 3 (ENGDAT V3).

- In allen 3 Versionen beginnt der Dateiname mit ENG. ENGDAT V3 erlaubt auch den Anfang mit EN3.
- Der Zeitstempel ist in den Versionen 1 und 2 genau 12 Zeichen lang und besitzt die folgende Struktur: JJMMTThhmmss. In der Version 3 ist er nur 10 Zeichen lang und hat eine andere Struktur: JTTThhmmss.

- Die freie Kennung (free reference code) besteht in allen ENGDAT-Versionen aus 5 Zeichen (alphanumerisch) und wird jeweils zwischen Sender und Empfänger bilateral vereinbart. Dieses Feld muss vollständig gefüllt werden.
- Die Dateianzahl (Anzahl der Dateien in einem ENGDAT-Paket) ist in den Versionen 1 bis 2 genau 3-Zeichen lang, wohingegen sie in der Version 3 genau 4-Zeichen lang ist.
- Sequenznummer (Nummer der Datei in einem ENGDAT-Paket): Es gilt das Gleiche wie für die Dateianzahl: in den Versionen 1 und 2 genau 3-Zeichen lang, wohingegen in der Version 3 genau 4-Zeichen lang. Die Dateien in einem ENGDAT-Paket werden durchnummeriert; die Abstract-Datei erhält immer die Nummer 1 (entspricht 001 in den Versionen 1 bis 2 und 0001 in der Version 3).

Beispiel für ENGDAT-Dateinamen (verschiedene Versionen):

Versionen 1 und 2: `ENG950730123305F1HBF004001`

Version 3: `ENG7115123305F1HBG00040002`

Der erste Teil des ENGDAT-Dateinamens bestehend aus Zeitstempel und freien Kennung wird auch Austauschreferenz oder Exchange Reference genannt: `ENGEr{Zeitstempel}{freie Kennung}`. Die Austauschreferenz muss die eindeutige Identifikation einer ENGDAT-Nachricht sicherstellen.

Im obigen Beispiel ist dies für die Versionen 1 und 2: `ENG950730123305F1HBF`. Es handelt sich um ein ENGDAT-Paket, welches am 30.07.95 um 12:33:05 erstellt wurde. Die freie Kennung lautet `F1HBF`; das Paket beinhaltet 4 Dateien (die Dateianzahl ist 004) und die Datei aus dem obigen Beispiel ist die erste (001 ist die Abstract-Datei). Für die Version 3 lautet der erste Teil: `ENG7115123305F1HBG`, davon ist `7115123305` der Zeitstempel, in dem 7115 den 115-ten Tag im Jahr 2007 darstellt und `123305` die Uhrzeit der Paketerstellung. Die Dateianzahl ist 0004, aber es handelt sich um die zweite Datei (0002) z.B. eine CAD-Datei, keine Abstrakt-Datei.

Alle Dateien mit dem gleichem ersten Teil bilden eine ENGDAT-Nachricht (oder ENGDAT-Paket).

Beispiel für ein ENGDAT-Paket (Version 1 bis 2):

Abstract-Datei: `ENG950708134537ABCDE003001`

CAD-Datei: `ENG950708134537ABCDE003002`

Textdatei: `ENG950708134537ABCDE003003`

Die freie Kennung (hier: ABCDE) ist wählbar und kann als Weiterleitungsadresse genutzt werden:

Die ENGDAT-Abstract-Datei besteht aus Nachrichtensegmenten, die in den verschiedenen Versionen unterschiedlich strukturiert sind.

Hinweis: Eine ausführliche Beschreibung der Nachricht und Syntax ist auf Anfrage erhältlich.

Interessante Links:

http://www.odette.org/html/freedownloads.htm

http://www.sasig.com/site/

Welche Segmente und Elemente tatsächlich verwendet werden, muss zwischen den Partnern vereinbart werden. Hierzu geben einige Automobilhersteller spezielle Richtlinien heraus, in denen die genaue Anwendung der ENGDAT-Nachricht beschrieben ist.

7.2 ENGDAT-Modul in rvsEVO

Folgende Funktionalitäten stehen Ihnen im ENGDAT-Modul zur Verfügung:

– empfangene ENGDAT-Nachrichten (ENGDAT-Pakete) anzeigen
– ENGDAT-Nachrichten zusammenstellen, anzeigen und versenden
– neue Dokumente einem schon vorhandenen ENGDAT-Paket hinzufügen oder Dokumente aus ihm entfernen (Hinweis: Ein ENGDAT-Paket besteht aus mindestens 2 Dateien: einer CAD/CAM-Datei und einer abstrakten Datei).
– ENGDAT-Partner verwalten.

Das Modul ENGDAT ist in rvsEVO über die Funktionsleiste, Symbol ENGDAT zu erreichen.

Wenn Sie auf dieses Symbol klicken, öffnet sich das ENGDAT-Fenster. Im linken Bereich des Fensters sehen Sie den Baum mit möglichen ENGDAT-Funktionen: ENGDAT-Umgebungsparameter; ENGDAT-Stammdaten: Sender, Empfänger, Kontakte, Stationen; Senden und Empfangen.

Normalerweise werden die ENGDAT-Umgebungsparameter bei der Installation auf Standardwerte gesetzt. Wenn Sie diese anpassen möchten, können Sie es über die Funktion **ENGDAT-Umgebungsdaten** tun.

Über die Funktion **ENGDAT - Stammdaten** lassen sich ENGDAT-Stationen, -Kontakte, -Sender und -Empfänger anlegen, bearbeiten und löschen.

Die Funktion **Senden** ermöglicht Ihnen neue ENGDAT-Pakete erstellen, senden, anzeigen und löschen.

Mit **Empfangen** können im Verzeichnis `$RVSPATH/engdat/files/in` empfangene ENGDAT-Pakete angezeigt, gelöscht oder kopiert werden.

7.3 ENGDAT-Stammdaten

Im Untereintrag **ENGDAT-Stammdaten** werden die Partnerdaten verwaltet. Die ENGDAT-Nachrichten werden in der Regel zwischen konkreten Personen (Ingenieure, Konstruteure, Techniker, etc) oder

Abteilungen in den verschiedenen Unternehmen ausgetauscht. Dabei sind die Sender die Personen aus der eigenen Firma und die Empfänger solche aus den Partnerfirmen. Jeder Partner (Sender ode Empfänger) identifiziert sich über die folgenden Daten:

- Kontaktinformationen - Name, Adressen, Firma, etc.
- Weiterleitungsadresse (Routing Code) - stellt die Verbindung zwischen der ENGDAT-Konfiguration und einer rvsEVO-Station her; jede Weiterleitungsadresse ist konkret einer rvsEVO-Station zugeordnet.
- Freie Kennung für den Dateinamen.

Außerdem wird für jeden Partner die ENGDAT-Version definiert, damit beim Versand von ENGDAT-Nachrichten sowohl die Dateinamen als auch die Abstract-Datei in dem Format erzeugt werden, welches von den jeweiligen Partner unterstützt wird.

Im Untereintrag **ENGDAT-Stammdaten** können Daten über ENGDAT-Sender, -Empfänger und über ENGDAT-Stationen, welchen die Sender und Empfänger zuzuordnen sind, verwaltet werden. Ausführliche Informationen über Partner (Sender und Empfänger) können unter **ENGDAT-Kontakte** abgelegt werden. Zu jeder Station können mehrere Sender (für die lokale Station) oder Empfänger (für die Partnerstation) konfiguriert werden. Die Verbindung zwischen einem Sender oder Empfänger und einer Station wird durch das Feld `Routing Code` realisiert.

In den Kontaktdaten (**ENGDAT- Kontakte**) sind ausführliche Informationen über Kontaktpersonen für das Senden oder Empfangen hinterlegt. Jeder Sender oder Empfänger hat eine eindeutige Id. Die Verbindung zwischen einem Sender oder Empfänger und dem zugehörigen Kontakt erfolgt über das Feld `Ingenieur Kontakt` (siehe Sender oder Empfänger-Maske).

Die Vorgehensweise beim Einrichten der ENGDAT-Stammdaten ist die Folgende:

– Zuerst sollen ENGDAT-Stationen eingerichtet werden.

– Danach werden Informationen über verschiedene Kontakpersonen eingepflegt.

– Nach dem Konfigurieren der ENGDAT-Kontakte und -Stationen sollen ENGDAT-Sender und -Empfänger eingerichtet werden. Die Verbindung von einer Station zu den jeweiligen Sendern/Empfängern wird über das Feld `Routing Code` hergestellt, wohingegen die Verbindung zwischen einem Kontakt und dem zugehörigen Sender/Empfänger über das Feld `Ingenieur Kontakt` erfolgt.

– Um ein neues ENGDAT-Paket zu versenden, verwenden Sie Funktion **Senden**. Mit der rechten Maustaste und anschließendem Anklicken, bekommen Sie das Fenster **Neues ENGDAT-Paket erstellen**. Mehr Informationen darüber, wie Sie ein ENGDAT-Paket versenden, finden Sie im Kapitel 7.7.

– Die empfangenen Dateien, die ins Verzeichnis `$RVSPATH/engdat/files/in` angekommen sind, können über die Funktionalität **Empfangen** angezeigt, gelöscht oder kopiert werden. Mehr Informationen darüber finden Sie im Kapitel 7.8.

7.4 ENGDAT-Stationen anlegen

Im rvsEVO-ENGDAT-Modul ist es notwendig, Stationen für die ENGDAT-Benutzung anzulegen. Die Zuordnung zwischen einem ENGDAT-Partner und der rvsEVO-Station wird mit dem ENGDAT-Feld `Routing Code` (Weiterleitungsadresse) gewährleistet.

Dieses Feld finden Sie als obligatorisches Feld in den ENGDAT-Segmenten SDE (Sender Engineering Contact) und RDE (Receiver Engineering Contact). Einer Station können mehrere Routing Codes und somit ENGDAT-Partner (Sender und Empfänger) zugeordnet werden.

Als Standardeinstellung sehen Sie nach dem ersten Start eines neu installierten rvsEVO die Stationen, die in rvsEVO schon vorhanden sind.

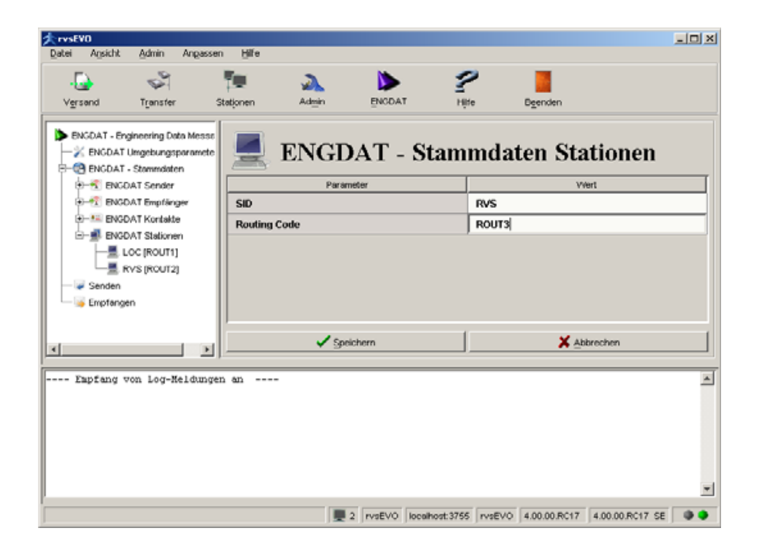
Wenn Sie eine neue Partnerstation für ENGDAT-Nachrichten brauchen, muss diese Station zuerst mit rvsEVO (`Admin -> Stationen`) angelegt werden.

Anschließend soll diese Station als ENGDAT-Station konfiguriert werden. Folgende Schritte sind notwendig, um eine neue ENGDAT-Station anzulegen:

– klicken Sie auf **ENGDAT-Stationen** mit der rechten Maustaste: Es öffnet sich das Kontextmenü **Neue Station erstellen**.

– Im rechten Teil des Fensters bekommen Sie dann die Möglichkeit unter `SID` die StationsID einer in rvsEVO schon vorhandenen Station auszuwählen.

– Das Feld `Routing Code` ist die Referenz für die Zuordnung einer Station zum jeweiligen Sender/Empfänger (Feld `RoutingCode` befindet sich in den ENGDAT-Segmenten SDE und RDE). Eine Station kann mehreren, unterschiedlichen RoutingCodes zugeordnet werden. Dazu muss diese Station neu angelegt werden (mit der gleichen SID, aber einem anderen RoutingCode. z.B. RVS[ROU2], RVS[ROU3]).



In diesem Beispiel wird der Station RVS einmal RoutingCode ROU2 und ein anderes Mal ROU3 zugeordnet.

7.5 ENGDAT-Kontakte anlegen

Nach dem Einrichten der Stationen ist es notwendig, ENGDAT-Kontakte (Informationen über Kontaktpersonen: Sender und Empfänger) einzupflegen. Über das Feld `interne Id Nummer` findet der jeweiligen Sender oder Empfänger die Verbindung zu den Kontaktdaten. Diese Id muss innerhalb der Sender- oder Empfängergruppe eindeutig sein. Da die restlichen Felder, die hier zu konfigurieren sind, selbsterklärend sind, wird in diesem Kapitel auf Ihre Beschreibung verzichtet.

In anderer Richtung gilt: Über das Feld `Ingenieur Kontakt` bei Sendern/Empfängern wird die Verbindung zum jeweiligen Kontakt hergestellt.

7.6 ENGDAT-Sender/-Empfänger anlegen

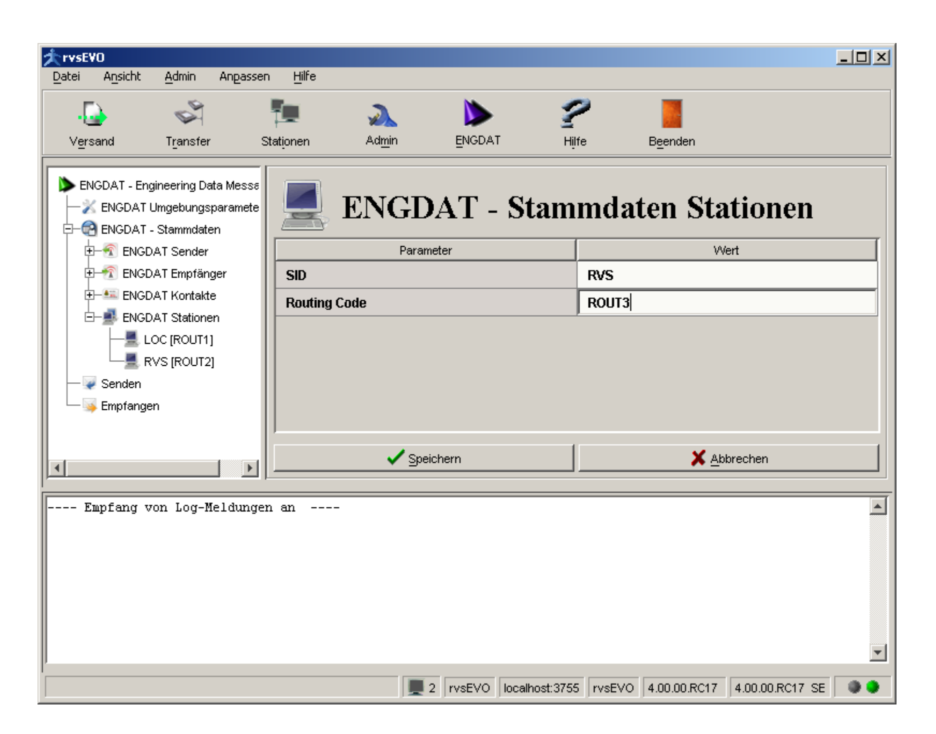
Ein rechter Klick auf **ENGDAT - Sender** bietet Ihnen die Funktion **Neuen Sender erstellen**. Im rechten Teils des Fensters erhalten Sie dann die Maske zum Erstellen eines neuen Senders. Die Felder, die zu konfigurieren sind, sind die Felder, die in der ENGDAT-Norm vorgeschrieben sind.

Folgende Felder sind zu konfigurieren:

`id`	interne eindeutige ID für die Verwaltung der Sender; nicht zu verwechseln mit der internen ID des ENGDAT-Kontakts

`Ingenieur Kontakt`	Hier ist die interne ID des ENGDAT-Kontakts anzugeben (siehe ENGDAT-Kontakte, interne id).
`Routing Code`	`Routing Codes` werden beim Anlegen einer ENGDAT-Station vergeben.
`Referenz Code`	`Referenz Code` ist bilateral zwischen Sender und Empfänger festzulegen, dieser 5-Zeichen lange alphanumerische Code taucht wieder im Namen einer ENGDAT-Datei auf. Siehe Kapitel 7.1 zur Erklärung, wie ein ENGDAT-Dateiname aufgebaut ist.
`In Verzeichnis`	Input-Verzeichnis für ENGDAT-Dateien. Standard: `$RVS_HOME/engdat/files/in`. In dieses Verzeichnis werden empfangene Dateien gespeichert.
`Out Verzeichnis`	Output-Verzeichnis für ENGDAT-Dateien. Standard: `$RVS_HOME/engdat/files/out`. In diesem Verzeichnis sollen die zu sendenden ENGDAT-Dateien, bereitgestellt werden.
`Engdat Version`	Die Engdat-Version, die der Partner unterstützt. Mögliche Werte: 1, 2 oder 3. Siehe Kapitel 7.1 für mehr Informationen über die ENGDAT-Norm.

Analog zum Erstellen von Sendern werden auch Empfänger erstellt. Mit der rechten Maustaste bekommen sie die Funktion **Neuen Empfänger erstellen**. Daraufhin öffnet sich im rechten Teil des Fenster die Maske, in der sie neue Empfänger-Parameter konfigurieren können.



Folgende Parameter sind zu konfigurieren:

`id`	interne eindeutige ID für die Verwaltung der Empfänger, nicht zu verwechseln mit der internen ID des Kontakts.
`Ingenieur Kontakt`	Hier ist die interne ID des Kontakts, der für diesen Empfänger zuständig ist, anzugeben (siehe Kontakte, interne ID).
`Routing Code`	Routing Codes werden beim Anlegen einer ENGDAT-Station vom Benutzer vergeben.
`Referenz Code`	`Referenz Code` ist bilateral zwischen Sender und Empfänger festzulegen; dieser 5-Zeichen lange alphanumerische Code ist wieder im Namen einer ENGDAT-Datei zu finden. Siehe Kapitel 7.1 zur Erklärung, wie ein ENGDAT-Dateiname aufgebaut ist.
`Engdat Version`	Die Engdat-Version, die der Partner unterstützt. Mögliche Werte: 1, 2 oder 3. Siehe Kapitel 7.1 für mehr Informationen über die ENGDAT-Norm.

7.7 ENGDAT-Pakete erstellen und versenden

Wenn Sie mit der rechten Maustaste auf **Senden** klicken, bekommen Sie zur Auswahl den Menüpunkt **Neues ENGDAT-Paket erstellen**. Mit dem Auswählen dieser Option öffnet sich im rechten Bereich ein neues Fenster, in dem Sie die Parameter für das Erzeugen eines neues ENGDAT-Pakets eingeben können.

In den Feldern Sender und Empfänger lassen sich schon angelegte Sender und Empfänger auswählen.

Folgende Namenskonvention wird dabei angewendet:

Interne ID des Senders oder Empfängers/ReferenzCode/RoutingCode (Bsp. 1/PW111/1)

Die Datei, die versendet werden soll, ist mit der Schaltfläche **Wähle Datei** auszuwählen.

Die restlichen Angaben sind optional. Die Parameter `Original Dateiname`, `Physischer Dateiname` und `Freier Text` sind aus dem ENGDAT-Segment EFC, wohingegen die Parametergruppe `Link zu Datei` zum LOF(Link to Other Files)-Segment gehört.

In der folgenden Tabelle finden Sie die Erklärung der Sende-Parameter:

Wähle Datei	Datei auf der Platte auswählen
Original Dateiname	Wenn keine Angabe, wird der Datename aus dem Feld **Wähle Datei** genommen (EFC-Segment).
Physischer Dateiname	Angabe zum physischen Dateinamen (EFC-Segment)
Freier Text	Kommentar
Sequenznummer	Die Sequenznummer wird benutzt, um eine Verbindung zwischen der Datei, die in diesem Paket (EFC Segment) vorliegt und einer anderen Datei aus dem gleichen Paket aber mit einem unterschiedlichen EFC Segment, herzustellen. Siehe ENGDAT V3 Dokumentation für mehr Informationen.
Link Beschreibung	freier Text, beschreibt den Zweck der Verbindung.

Hinweis: Solange die ENGDAT-Nachricht noch nicht versendet wurde, liegt die Abstract-Datei intern im ENGDAT V3 Format vor. Beim Bereitstellen zum Senden wird sie dann in ENGDAT V2 konvertiert, wenn für den Partner nicht ENGDAT V3 konfiguriert ist. ENGDAT V3 und V2 sind von der Nachrichtenstruktur nicht 100%ig identisch. Das bedeutet aber,

dass einige Informationen gemappt werden müssen, z.B. wird der original Dateiname in das Element 1896 im Segment EFC geschrieben. Der physische Dateiname wird als EFC Segment geschrieben, da es keine Entsprechung bei V2 gibt.

7.8 ENGDAT-Pakete empfangen

Um die ENGDAT-Pakete, die empfangen wurden, zu kopieren, anzuzeigen oder zu löschen, klicken Sie auf den Menüpunkt **Empfangen**. Im rechten Fenster erscheinen die Informationen über die schon empfangenen Dateien.

Folgende Informationen sind zu sehen: `Dokument Id`, `Dokument Datum`, `Version` (ENGDAT), `SDE Kontakt/RoutingCode`, `RDE Kontakt/RoutingCode`.

Die empfangenen Dateien befinden sich im Verzeichnis `$RVS_HOME/engdat/files/in`.

7.9 Überblick: ENGDAT-Verzeichnisse

In diesem Kapitel wird ein kurzer Überblick über die in `$RVS_HOME/engdat` vorhandenen Verzeichnisse gegeben.

Folgende Unterverzeichnisse sind im Verzeichnis `RVS_HOME/engdat` vorhanden:

– `bin`: Verzeichnis für Skripte

– `conf`: Verzeichnis für ENGDAT-Konfigurationsdateien. In diesem Verzeichnis befinden sich die Dateien: `engdat.properties`, `rvsEngPartners.xml` und `rvsxmlLogger.xml`. `engdat.properties` beinhaltet engdat-Konfigurationsparameter (`GUI -> Admin -> Parameter`). `rvsEngPartners.xml` beinhaltet Stammdaten von Sendern, Empfängern, Kontakten und Stationen. `rvsxmlLogger.xml` ist die interne Datei für Logging (Debug).

– `files/out`: Verzeichnis für zu versendende Dateien

– `files/in`: Verzeichnis für empfangene Dateien

– `log`: Verzeichnis für log-Dateien.

– `status`: Verzeichnisse für Job-Dateien. Im Unterverzeichnis `active` befinden sich die Dateien, die sich auf aktive Jobs beziehen. Die Dateien über abgeschlossene Jobs werden im Unterverzeichnis `archive` abgelegt.

– `system`: Verzeichnis für ENGDAT-Systemdateien.

8 Zentrale Administration von rvsEVO

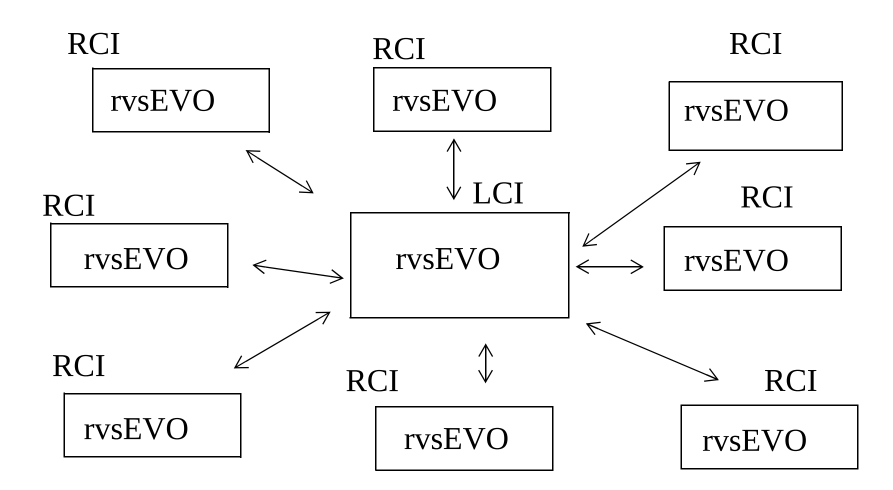
In diesem Kapitel ist eine mächtige Funktionalität von rvsEVO beschrieben: wie man mit einer rvsEVO-Installation andere rvsEVO Stationen administrieren kann. Die Zentrale Administration ist von besonderer Bedeutung für den rvsEVO-Netzwerk-Administrator, da sie ihm ermöglicht, alle rvsEVO-Installationen entfernt zu pflegen.

8.1 Einleitung

Die Zentrale Administration von rvsEVO ermöglicht das Administrieren von zahlreichen rvsEVO-Installationen mit Hilfe spezieller Konfigurationsdateien.

Die Konfigurationsdateien werden von der rvsEVO-Station gesendet, die andere rvsEVO-Stationen administriert (sie wird in diesem Buch Local Configuration Instance, LCI, genannt) zu der Station, die administriert weden soll (wir nennen sie: Remote Controlled Instance RCI). Die beiden Stationen (Instanzen) müssen in einem Stern-Netzwerk mit einander kommunizieren können (Siehe die folgende Abbildung).

Abbildung



Konfigurationsdateien

Die Konfigurationsdateien für die Zentrale Administration nennen wir **Container - Konfigurationsdateien**, um sie von den normalen rvsEVO XML-Konfigurationsdateien wie z.B. `rvsStationlist.xml` zu unterscheiden.

Container - Konfigurationsdateien sind:

– `cfg.req.jar` für Konfigurationsanfragen, die LCI an RCI sendet

– `cfg.rsp.jar` für Konfigurationsantworten, die RCI an LCI sendet.

Die Local Configuration Instance LCI

LCI

Jede rvsEVO-Installation kann eine LCI werden.

Es gibt ein spezielles Verzeichnis in rvsEVO, wo Konfigurationsdaten von allen RCIs (rvsEVO-Stationen, die entfernt administriert werden sollen) gespeichert werden - das ist das Configuration Repository CRep. Das CRep-Verzeichnis ist `$RVS_HOME/management`. Siehe bitte Kapitel 1.4 für eine detaillierte Erklärung von `$RVS_HOME`.

CRep

Das CRep (`$RVS_HOME/management`) besteht aus folgenden Verzeichnissen oder Dateien:

- `$RVS_HOME/management/mgmt-datastore`
  Dieses Verzeichnis entsteht nur nach einer erfolgreichen Übertragung und erhaltenen Antwort auf eine Anforderung (request). Die Anforderung einer Container-Konfigurationsdatei stellt eine LCI an eine RCI. Siehe bitte Kapitel 8.3 für die Erklärung, wie eine Anforderung erzeugt wird.
- `$RVS_HOME/management/mgmt-log/activity.log` Diese log-Datei enthält Protokolle aller Konfigurationsschritte. Jeder Eintrag in dieser Datei besteht aus einem Zeitstempel, dem Typ der Konfiguration: **configuration request** (Konfigurations - Anforderung) oder **configuration response** (Konfigurations - Antwort), der SID der administrierten rvsEVO-Station (RCI) und der Aktivitätsmeldung.
- `$RVS_HOME/management/mgmt-templates`
  Dieses Verzeichnis enthält templates für die Aktionen, die der rvsEVO-Netzwerk-Administrator durchführt. Sie brauchen dieses Verzeichnis nur, wenn Sie die Administrator-Software updaten.
- `$RVS_HOME/management/mgmt-workspace`
  In diesem Verzeichnis sind Konfigurationsdateien von RCIs, die administriert werden sollen, gespeichert. Das Verzeichnis enthält Unterverzeichnisse, die nach der RCIs SID benannt werden (Siehe bitte das Programm `prepareUpdateStation`, Kapitel 8.3 als Beispiel).

Die Remote Controlled Instance RCI

RCI

Eine RCI ist die rvsEVO-Station, die entfernt administriert werden soll.

Wenn eine rvsEVO-RCI eine Konfigurationsanforderung (Datei `cfg.req.jar`) von einer LCI empfängt, löst dieser Vorgang den Start eines entsprechenden Jobs aus. Dieser Job verarbeitet die Konfigurationsanforderung und generiert eine Konfigurationsantwort. Die Konfigurationsantwort wird dann an die LCI als Datei `cfg.rsp.jar` zurückgesendet.

8.2 Programme der Zentralen Administration

Programme

Der ganze Prozess (alle Konfigurations- und Administrationsfälle) sind mit den folgenden Programmen auszuführen. Diese Programme befinden sich im Verzeichnis `$RVS_HOME\bin` als Batch-Dateien (Siehe Kapitel 1.4 für eine detaillierte Erklärung von `$RVS_HOME`). Um sie ausführen zu können, müssen Sie sie aus dem Verzeichnis `$RVS_HOME\bin` starten.

orderConfiguration
-s <SID>

holt die Konfiguration von einer RCI und speichert diese ins CRep-Verzeichnis.

Beispiel (für CRep):
`C:\rvsEVO\management\mgmt-datastore\TINYPW`

In diesem Beispiel ist `TINYPW` die SID von der RCI (von der Station, die administriert werden soll).

prepareUpdateStation
-s <SID>

holt eine Kopie der RCI-Konfiguration aus dem CRep-Verzeichnis ins WorkDir, um dort die Konfiguration durchzuführen.

Beispiel für WorkDir:

Im Verzeichnis `C:\rvsEVO\management\mgmt-workspace` wird folendes Unterverzeichnis erzeugt: `TINYPW\UPDATE_STATION_040826_114418\out`.

Das ist das Unterverzeichnis für die Konfiguration der Station `TINYPW`; Datum der Erzeugung ist 2004-08-26, Zeit 11:44:18.

commitUpdateStation
-s <SID> -d <WorkDir>

sendet die modifizierte Konfiguration aus dem WorkDir (ohne Unterverzeichnis `out`) zur RCI (Die RCI soll in der Option `-s` für die StationsID angegeben werden).

Beispiel:

```
commitUpdateStation -s TINYPW
-d C:\rvsEVO\management\mgmt-
workspace\TINYPW\UPDATE_STATION
_040826_114418
```

Hinweis: Der Name des Verzeichnisses WorkDir besteht aus dem CRep (Siehe Kapitel 8.1); der SID der RCI (`TINYPW` im obigen Beispiel), dem Verzeichnis `UPDATE_STATION` mit dem Zeitstempel (der wiederum aus Datum und Uhrzeit besteht; im Beispiel `040826_114418`) und dem Verzeichnis `out`.

WorkDir enthält alle wichtigsten rvsEVO-Verzeichnisse und Dateien: Hier ist ein Beispiel, wie Sie die Konfigurationsdateien ändern können:

– Ändern Sie die Stationskonfiguration, indem Sie die Datei `$RVS_HOME/conf/rvsStationlist.xml` editieren.

– Ändern Sie die Jobstarts-Konfiguration, indem Sie die Datei `$RVS_HOME/conf/rvsJobstart.xml` editieren.

– Machen Sie ein Update der Software, indem Sie die .jar-Dateien im Verzeichnis `$RVS_HOME/lib` austauschen.

8.3 Wie arbeite ich mit der Zentralen Administration?

Die folgenden Schritte sollten Ihnen als ein typisches Konfigurationsbeispiel dienen. Alle diese Schritte sind notwendig, unabhängig davon, ob Sie ein Update durchführen, einen Lizenzschlüssel austauschen oder die Konfiguration der Stationen, Parameter oder Jobstarts ändern möchten. Mehr Details über derartige Administrationsaufgaben erhalten Sie in den Kapiteln 8.3.1, 8.3.2 und 8.3.3.

Hinweis: Die folgenden Befehle sind als Batch-Dateien im Verzeichnis `$RVS_HOME\bin` vorhanden (Siehe bitte Kapitel 1.4 für eine detaillierte Erklärung von `$RVS_HOME`), so dass Sie sich in diesem Verzeichnis befinden müssen, um diese Programme ausführen zu können.

Schritte

- Der erste Schritt ist immer, die Konfiguration der zu administrierenden Station, RCI, anzufordern. Für diesen Zweck dient das Programm `orderConfiguration`.
  Beispiel: Wenn Sie die Konfiguration der rvsEVO Station `TINY11` ändern möchten, müssen Sie in der Eingabeaufforderung (Kommandozeile) das Programm `orderConfiguration` mit dem folgenden Befehl starten:

  ```
  orderConfiguration -s TINY11
  ```

  Dieser Befehl erzeugt die Konfigurationsanforderungsdatei `cfg.req.jar` (Siehe Kapitel 8.1 für die Erklärung der `cfg.req.jar`) und sendet diese Datei via OFTP zur Station `TINY11`. Die Übertragung der Datei `cfg.req.jar` können Sie in der rvsEVO GUI (Admin - Fenster) verfolgen. Mit dem Empfang der Datei `cfg.req.jar` bei der RCI (`TINY11`), wird ein Konfigurationsprozess angestoßen. Dieser Prozess stoppt rvsEVO (bei der Station `TINY11`), archiviert die aktuelle Konfiguration in der Datei `cfg.rsp.jar`, startet rvsEVO (`TINY11`) wieder und sendet die Konfigurationsantwortdatei `cfg.rsp.jar` zurück zur LCI (rvsEVO-Station des Administrators). Im Falle, dass dieser Schritt erfolgreich war, bekommen Sie in der Eingabeaufforderung die Meldung „OrderConfiguration exited with return code *0*“. Alle diese Schritte führt das Programm `orderConfiguration` automatisch aus.

  Der nächste Schritt ist das Kopieren der RCI-Konfiguration, die als Datei `cfg.rsp.jar` empfangen wurde. Das ist die Aufgabe des Programms `prepareUpdateStation`. Dieses Programm kopiert für Sie die empfangenen Konfigurationsantwortdatei `cfg.rsp.jar` und speichert diese ins WorkDir - Verzeichnis (Siehe Kapitel 8.2 für die Erklärung von WorkDir).

  Beispiel:

```
prepareUpdateStation -s TINY11
```

Ergebnis: Das Verzeichnis C:\rvsEVO\management\mgmt-workspace\TINY11\UPDATE_STATION_040828_113315\out mit der kompleten Konfiguration der Station TINY11 wird erzeugt. Die Meldung über den Erfolg oder Misserfolg dieses Schrittes finden Sie in der Datei activity.log. Diese log - Datei befindet sich im Verzeichnis $RVS_HOME/management/mgmt-log.

- Jetzt können Sie die Konfiguration der RCI (rvsEVO-Station TINY11 im Beispiel) ändern. Sie können z.B. einen Lizenzschlüssel austauschen, die XML-Konfigurationsdatei für Stationen oder Jobstarts editieren oder durch das Ersetzen der ensprechenden .jar-Datei ein Update von rvsEVO durchführen. Mehr Details über derartige Administrationsaufgaben erhalten Sie in den Kapiteln 8.3.1, 8.3.2 und 8.3.3).
- Senden Sie die modifizierte Konfiguration zur RCI (TINY11). Sie müssen das ganze Verzeichnis C:\rvsEVO\management\mgmt-workspace\TINY11\UPDATE_STATION_040828_113315) mit dem folgenden Befehl senden:

```
commitUpdateStation -s TINY11 -d
C:\rvsEVO\management\mgmt-
workspace\TINY11\UPDATE_STATION_040828_113315
```

Dieser Befehl wird das modifizierte Verzeichnis zusammenfassen und zur RCI (TINY11) unter dem Dateinamen cfg.req.jar zurücksenden.

Nachdem die RCI-Station (TINY11) die Datei cfg.req.jar erhalten hat, wird rvsEVO gestoppt (alle diese Schritte werden vom Programm commitUpdateStation ausgeführt, Sie brauchen keine zusätzlichen Aktionen zu starten) und die geänderte Konfiguration wird aktualisiert. Der Job prüft zusätzlich, ob alles fehlerfrei durchgeführt wurde und sendet die Datei cfg.rsp.jar als Antwort zurück. Das Ergebnis dieser Aktion wird auch in der Datei activity.log protokolliert.

Hinweis: Im Falle eines Misserfolgs wird die alte Konfiguration wieder aktiviert.

8.3.1 Wie tausche ich einen Lizenzschlüssel aus?

Lizenzschlüssel

Dieses Kapitel beschreibt einen typischen Fall bei der Administration von rvsEVO-Stationen, nämlich wie man einen nicht mehr gültigen Lizenzschlüssel austauscht. Im nächsten Beispiel wird die Station TINY01 die Station TINY02 administrieren.

Voraussetzungen: Die Station TINY01 muss die Station TINY02 in der Stationstabelle als eine Nachbarstation haben (Siehe bitte Kapitel 3.2.1 für die Erklärung, wie man Stationen anlegt) und die Station TINY02

muss die Station TINY01 als als eine Nachbarstation in der Stationsliste haben.

Schritte

- Der erste Schritt, um einen Lizenzschlüssel auszutauschen, ist mit dem folgenden Befehl in der Eingabeaufforderung die Konfiguration von TINY02 anzufordern:

  ```
  orderConfiguration -s TINY02
  ```

  Bei einem Erfolg dieser Aktion, bekommen Sie in der Eingabeaufforderung die Meldung „OrderConfiguration exited with return code *0*" und anschließend werden Sie als Antwort auf Ihre Konfigurationsanforderung die Datei cfg.rsp.jar erhalten. (Das können Sie bei den **Beendeten Übertragungen** im Admin-Fenster von der TINY01 GUI verfolgen.

- Der nächste Schritt ist, eine Kopie der TINY02-Konfiguration zu erzeugen (diese Konfiguration ist in der Datei cfg.rsp.jar enthalten). Dazu wird der folgende Befehl benutzt:

  ```
  prepareUpdateStation -s TINY02
  ```

  Ergebnis: Das Verzeichnis C:\rvsEVO\management\mgmt-workspace\TINY02\UPDATE_STATION_040828_113315\out mit der vollständigen Konfiguration der Station TINY02 wurde erzeugt. Das Ergebnis dieser Aktion wird in der Datei activity.log protokolliert. Diese log - Datei befindet sich im Verzeichnis $RVS_HOME/management/mgmt-log.

- Jetzt können Sie den alten Lizenzschlüssel aus dem Verzeichnis C:\rvsEVO\management\mgmt-workspace\TINY02\UPDATE_STATION_040828_113315\out\conf nach licenseOLD.properties umbenennen und den neuen Lizenzschlüssel license.properties ins Verzeichnis C:\rvsEVO\management\mgmt-workspace\TINY02\UPDATE_STATION_040828_113315\out\conf kopieren. Wie Sie einen neuen Lizenzschlüssel erwerben können, lesen Sie bitte im Kapitel 2.2.

- Senden Sie die geänderte Konfiguration zur Station TINY02 (Sie müssen das ganze Verzeichnis C:\rvsEVO\management\mgmt-workspace\TINY02\UPDATE_STATION_040828_113315) mit dem folgenden Befehl senden:

  ```
  commitUpdateStation -s TINY02 -d
  C:\rvsTiny\management\mgmt-
  workspace\TINY02\UPDATE_STATION_040828_113315
  ```

  Dieser Befehl wird das ganze geänderte Verzeichnis zusammenfassen und unter dem Dateinnamen cfg.req.jar an die Station TINY02 senden.

- Nach erfolgreichem Empfang der Datei `cfg.req.jar` bei der Station `TINY02`, wird rvsEVO von `TINY02` gestoppt (alle diese Schritte werden vom Programm `commitUpdateStation` selbständig ausgeführt); die geänderte Konfiguration wird aktualisiert; der Job prüft noch, ob alles fehlerfrei abgelaufen war und sendet die Antwort unter dem Dateinamen `cfg.rsp.jar` zurück. Das Ergebnis dieser Änderungen wird in der Datei `activity.log` protokolliert.

Hinweis: Im Falle eines Misserfolgs wird auf der RCI die alte Konfiguration wieder eingespielt.

8.3.2 Wie ändere ich die Konfiguration einer Station?

Konfiguration einer Station

Dieses Kapitel beschreibt einen typischen Fall in der Administration von rvsEVO, nämlich wie man Stationsparameter z.B. die ODETTE-ID ändern kann. Die gleiche Vorgehensweise gilt auch für andere Parameter (andere Stationsparameter, Jobstartparameter oder rvsEVO globale Parameter). In diesem Beispiel wird die Station `TINY20` die Station `TINY22` administrieren.

Voraussetzungen: Die Station `TINY20` muss die Station `TINY22` in der Stationstabelle haben (Siehe bitte Kapitel 3.2.1 für die Erklärung, wie man Stationen anlegt) und die Station `TINY22` muss die Station `TINY20` als eine Nachbarstation in der Stationstabelle haben.

Schritte

- Um die Station von `TINY22` entfernt konfigurieren zu können, muss die Station `TINY20` zuerst die Konfiguration von `TINY22` anfordern. Das geschieht mit dem folgenden Befehl:

  ```
  orderConfiguration -s TINY22
  ```

  Bei einem Erfolg dieser Aktion bekommen Sie in der Eingabeaufforderung die Meldung „OrderConfiguration exited with return code *0*“ und anschließend werden Sie als Antwort auf Ihre Konfigurationsanforderung die Datei `cfg.rsp.jar` erhalten. (Das können Sie bei den **Beendeten Übertragungen** im Admin - Fenster von der `TINY20` GUI verfolgen.

- Der nächste Schritt ist, eine Kopie der `TINY22` - Konfiguration im WorkDir - Verzeichnis von `TINY20` zu erzeugen (diese Konfiguration ist in der Datei `cfg.rsp.jar` enthalten). Dazu wird der folgende Befehl benutzt:

  ```
  prepareUpdateStation -s TINY22
  ```

  Ergebnis: Das Verzeichnis `C:\rvsEVO\management\mgmt-workspace\TINY22\UPDATE_STATION_040828_113315\out` mit der vollständigen Konfiguration der Station `TINY22` wurde erzeugt. Das Ergebnis dieser Aktion wird in der Datei `activity.log` protokolliert. Diese log - Datei befindet sich im Verzeichnis `$RVS_HOME/management/mgmt-log`.

- Jetzt können Sie die Datei `rvsStationlist.xml` aus dem Verzeichnis `C:\rvsEVO\management\mgmt-`

workspace\TINY22\UPDATE_STATION_040829_133315 \out\conf editieren und z.B. den Parameter ODETTE_ID oder TCP/IP - Adresse (Parameter IP_ADDR) für die lokale Station von TINY22 (STATION_LOC) oder für andere Stationen (Nachbarstationen oder geroutete Stationen) ändern.

- Der nächste Schritt ist, die geänderte TINY22-Konfiguration an diese Station zu senden. Sie müssen das ganze Verzeichnis (aber ohne Unterverzeichnis out) C:\rvsEVO\management\mgmt-workspace\TINY02\UPDATE_STATION_040829_133315) mit dem folgenden Befehl senden:

  ```
  commitUpdateStation -s TINY22 -d
  C:\rvsEVO\management\mgmt-
  workspace\TINY22\UPDATE_STATION_040829_133315
  ```

  Dieser Befehl wird das ganze modifizierte Verzeichnis zusammenfassen und unter dem Dateinamen cfg.req.jar an die Station TINY22 senden.

- Nach dem erfolgreichen Empfang der Datei cfg.req.jar bei der Station TINY22, wird rvsEVO auf TINY22 gestoppt (alle diese Schritte werden vom Programm commitUpdateStation selbständig ausgeführt); die geänderte Konfiguration wird aktualisiert; der Job prüft noch, ob alles fehlerfrei abgelaufen war und sendet die Antwort unter dem Dateinamen cfg.rsp.jar zurück. Das Ergebnis dieser Änderungen wird auf der LCI in der Datei activity.log protokolliert.

Hinweis: Im Falle eines Misserfolgs wird auf der RCI die alte Konfiguration wieder eingespielt.

8.3.3 Wie führe ich ein Update von rvsEVO durch?

Update

Dieses Kapitel beschreibt einen typischen Fall in der Administration von rvsEVO-Stationen, nämlich wie man ein Update einer anderen rvsEVO - Station durchführt. In diesem Beispiel wird die Station TINY30 die Station TINY33 administrieren.

Voraussetzungen: Die Station TINY30 muss die Station TINY33 in der Stationstabelle haben (Siehe bitte Kapitel 3.2.1 für die Erklärung, wie man Stationen anlegt) und die Station TINY33 muss die Station TINY30 in ihrer Stationstabelle haben.

Schritte

- Um die Station TINY33 updaten zu können, muss die Station TINY30 zuerst die Konfiguration von TINY33 anfordern. Das geschieht mit dem folgenden Befehl:

  ```
  orderConfiguration -s TINY33
  ```

  Bei einem Erfolg dieser Aktion bekommen Sie in der Eingabeaufforderung die Meldung „OrderConfiguration exited with return code *0*“ und anschließend werden Sie als Antwort auf Ihre Konfigurationsanforderung die Datei cfg.rsp.jar erhalten. (Das

können Sie bei den **Beendeten Übertragungen** im Admin - Fenster von der `TINY30` GUI verfolgen.

- Der nächste Schritt ist, eine Kopie der `TINY33 -` Konfiguration im WorkDir - Verzeichnis von `TINY30` zu erzeugen (diese Konfiguration ist in der Datei `cfg.rsp.jar` enthalten). Dazu wird der folgende Befehl benutzt:

  ```
  prepareUpdateStation -s TINY33
  ```

  Ergebnis: Das Verzeichnis `C:\rvsEVO\management\mgmt-workspace\TINY33\UPDATE_STATION_040830_113315\out` mit der vollständigen Konfiguration der Station `TINY33` wurde erzeugt. Das Ergebnis dieser Aktion wird in der Datei `activity.log` protokolliert. Diese log - Datei befindet sich im Verzeichnis `$RVS_HOME/management/mgmt-log`.

- Benennen Sie jetzt die alte .jar-Datei `rvs.jar` aus dem Verzeichnis `C:\rvsEVO\management\mgmt-workspace\TINY33\UPDATE_STATION_040830_113315\out\lib` nach `rvsOLD.jar` um und ersetzen Sie die alte .jar-Datei durch eine neue. Um eine aktuelle .jar-Datei für das Update zu bekommen, wenden Sie sich bitte an den rvs® - Kundendienst (E-Mail: rvs-service@gedas.de; Tel. +49 30 39971 777; fax: +49 30 39971 994).

- Senden Sie ist die geänderte `TINY33 -` Konfiguration zu dieser Station. Sie müssen das ganze Verzeichnis (aber ohne Unterverzeichnis `out`) `C:\rvsEVO\management\mgmt-workspace\TINY33\UPDATE_STATION_040829_133315`) mit dem folgenden Befehl senden:

  ```
  commitUpdateStation -s TINY33 -d
  C:\rvsEVO\management\mgmt-
  workspace\TINY02\UPDATE_STATION_040830_113315
  ```

  Dieser Befehl wird das ganze modifizierte Verzeichnis zusammenfassen und unter dem Dateinamen `cfg.req.jar` an die Station `TINY33` senden.

- Nach dem erfolgreichen Empfang der Datei `cfg.req.jar` bei der Station `TINY33`, wird rvsEVO auf `TINY33` gestoppt (alle diese Schritte werden vom Programm `commitUpdateStation` selbständig ausgeführt); die geänderte Konfiguration wird aktualisiert; der Job prüft noch, ob alles fehlerfrei abgelaufen war und sendet die Antwort unter dem Dateinamen `cfg.rsp.jar` zurück. Das Ergebnis dieser Änderungen wird auf der LCI in der Datei `activity.log` protokolliert.

Hinweis: Im Falle eines Misserfolgs wird auf der RCI die alte Konfiguration wieder eingespielt.